'68' MICRO JOURNAL

$2.50 USA

Australia	A $ 4.00	New Zealand	NZ $ 4.00
Singapore	S $ 8.00	Hong Kong	H $20.00
Malaysia	M $ 8.00		

VOLUME II ISSUE 2 • Devoted to the 68XX User • February 1980

"Small Computers Doing Big Things."

SERVING THE 6800 USERS WORLDWIDE

SYSTEMS – SOLUTIONS

If you have a problem that can be solved by a computer—we have a systems solution.

- Two central processors with maximum RAM capacities of 56K and 384 K bytes
- Three types of disk drives with capacities of 175K, 1.2M and 16M bytes
- Two dot matrix printers with 80 and 132 line capacity
- A Selectric typewriter interface and a daisy wheel printer

Match these to your exact need, add one or more of our intelligent terminals and put together a system from one source with guaranteed compatibility in both software and hardware.

Southwest Technical Products systems give you unmatched power, speed and versatility. They are packaged in custom designed woodgrain finished cabinets. Factory service and support on the entire system and local service is available in many cities.

SOUTHWEST TECHNICAL PRODUCTS CORPORATION
219 W. RHAPSODY
SAN ANTONIO, TEXAS 78216 (512) 344-0241

The Source for 6800/6809 Software

Technical Systems Consultants, Inc. is The Source for your 6800/6809 systems software needs. From FLEX™, the standard disk operating system of the 680X family, to Sort/Merge, your systems requirements can be filled with the highest quality software in the industry. Nowhere else can you find such variety from a single source. Here are some of the most popular:

Program Name	6800	6809
FLEX for SWTPc	$90	$90
FLEX for SSB	90	90
Extended BASIC	100	100
Extended BASIC Precompiler	50	50
BASIC	65	65
BASIC Precompiler	40	40
FLEX Sort/Merge	75	75
Text Editing System	40	35
Assembler	40	40
Text Processing System	60	N/A
Debug Package	55	75
FLEX Utilities	100	60

These packages are available on either 8" or 5" soft-sectored FLEX diskettes (5" 6800 is FLEX 2.0). Price includes user's manual and object code diskette. Certain programs are available on cassette. Contact Technical Systems Consultants for pricing. All orders should include 3 percent for postage and handling (8 percent on foreign orders). Master Charge and Visa are welcome.

™FLEX is a trademark of Technical Systems Consultants, Inc.

technical systems consultants, inc.
Box 2574, West Lafayette, IN 47906
(317) 463-2502

'68' MICRO JOURNAL

Portions of the text of '68' Micro Journal set using the following
6800/2, DMAF1 and CT-82
Southwest Technical Products Corp.
219 W. Rhapsody
San Antonio, TX 78216

Editor, Word Processor and Sort Merge
Technical Systems Consultants, Inc.
Box 2574
W LaFayette, IN 47906
'MINIFLEX & FLEX REG ™ Technical Systems Consultants, Inc

Selectric I/O
World Wide Electronics, Inc.
130 Northwestern Blvd
Nashua, NH 03060

Publisher/Editor
Don Williams Sr.

Executive Editor
Larry E. Williams

Assistant Editor — Hardware
Dennis Womack

Associate Editor — Southwest
Dr. Jack Bryant

Associate Editor — At Large
Dr. Chuck Adams

Associate Editor — Midwest
Howard Berenbon

Contributing Editors
Dr. Jeffrey E. Brownstein
Dale Puckett
T. Jackson — Japan
Russell Gore

Subscriptions and Office Manager
Joyce Williams

Typography and Color Separations
Williams Company, Inc.
Chattanooga, TN 37421

** CONTENTS **

Send All Correspondence To:

'68' Micro Journal
3018 Hamill Rd.
PO Box 849
Hixson, Tennessee 37343

— Phone —
Office: 615-870-1993
Plant: 615-892-7544

'68' Micro Journal is published 12 times a year by '68' Micro Journal, 6131 Airways Blvd., Chattanooga, TN 37421. Second Class postage paid at Chattanooga, TN. Postmaster: Send Form 3579 to '68' Micro Journal, PO Box 849, Hixson, TN 37343.

1-Year $14.50 2 Years $26.00 3 Years $36.50

—ITEMS SUBMITTED FOR PUBLICATION—

(Letters to the Editor for Publication) All 'letters to the Editor' should be substantiated by facts. Opinions should be indicated as such. All letters must be signed. We are interested in receiving letters that will benefit or alert our readers. Praise as well as gripes is always good subject matter. Your name may be withheld upon request. If you have had a good experience with a 6800 vendor please put it in a letter. If the experience was bad put that in a letter also. Remember, if you tell us who they are then it is only fair that your name 'not' be withheld. This means that all letters published, of a critical nature, cannot have a name withheld. We will attempt to publish 'verbatim' letters that are composed using 'good taste.' We reserve the right to define (for '68' Micro) what constitutes 'good taste.'

(Articles and items submitted for publication) Please, always include your full name, address, and telephone number. Date and number all sheets. TYPE them if you can, poorly handwritten copy is sometimes the difference between go, no-go. All items should be on 8X11 inch, white paper. Most all art work will be reproduced photographically, this includes all listings, diagrams and other non-text material. All typewritten copy should be done with a NEW RIBBON. All hand drawn art should be black on white paper. Please no hand written code items over 50 bytes. Neatly typed copy will be directly reproduced. Column width should be 3¼ inches.

(Advertising) Any Classified: Maximum 20 words. All single letters and/or numbers will be considered one (1) word. No Commercial or Business Type Classified advertising. Classified ads will be published in our standard format. Classified ads $7.50 one time run, paid in advance.

Commercial and/or Business advertisers please write or phone for current rate sheet and publication lag time.

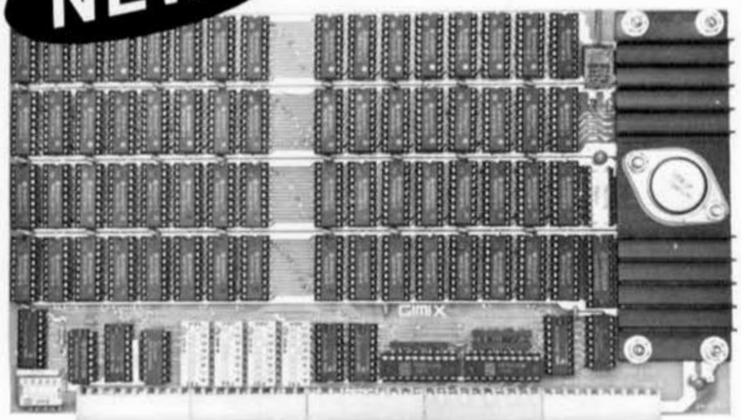
WANT TO BE KING OF THE HILL?

TREAT YOURSELF ROYALLY WITH GIMIX UNIQUE AND INCOMPARABLE BOARDS AND SYSTEMS... DIP-switch Versatility for use with both SS50 (6800) and SS50C (6809) Systems (SWTP. etc.)

THE FIRST AND ONLY 32K STATIC RAM BOARD...

NEW!

Designed for use with: ★ Existing SS50 Systems
★ SS50C Extended Address Systems



FEATURES:

- Decoding for 4 Extended Address Lines (allows memory decoding up to 1 megabyte)
- DIP-switch to set extended addressing or disable it
- 4 separate 8K blocks, addressable to any 8K boundary by DIP-switch
- Each 8K block may be individually disabled
- Write protect either of two 16K sections
- Low power consumption — uses 2114L low power RAMS — (2 amps typical for 32K)
- Fully Socketed
- Gold Bus Connectors

Assembled, Burned In and Tested at 2MHz.

16K .... $328.12 24K .... $438.14 32K $548.15

16K and 24K Versions are socketed for 32K and require only additional 2114S for expansion.

FACTORY PRIME STATIC RAMS		
	2114L 450 ns $5.90	200 ns $6.90
	4044 450 ns $5.90	250 ns $6.90

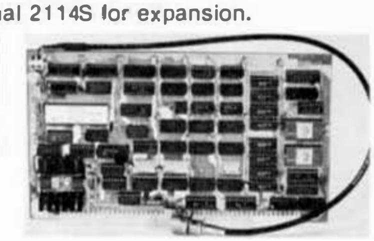
ADD $5.00 HANDLING ON ORDERS UNDER $200.00

THE UNIQUE GIMIX 80 x 24 VIDEO BOARD

★ Upper and Lower Case with Descenders ★ Hardware Scrolling
★ Contiguous 8x10 Character Cells ★ X-Y Addressable Hardware Cursor

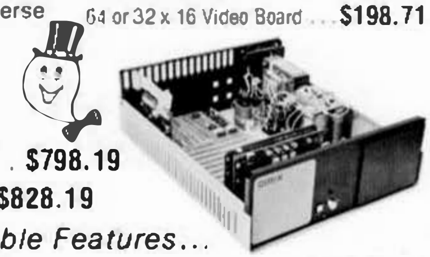
It is the ***ONLY*** **Video Board** that gives you:

- A user programmable RAM character generator. Custom character sets, up to 128 characters each, can be stored and loaded into the board *under software control*, from disk, tape, etc.
- The ability to choose, *under software control*, 256 displayable characters from 384 available in the 3 on board (2 EPROM and 1 RAM) character generators.
- The ability to divide the 256 displayable characters into 8 groups, according to *both* ASCII Code and bit 8; lets your program determine how each group is displayed. (Which character generator to use, and whether it will be normal or inverse video, full or reduced intensity or a combination of these.)
- GHOSTability: to place multiple boards at the same address and access them individually without affecting the display of the other boards.
- The ability to control all these features, on the fly, through software.



Fully decoded, occupies only 2K of address space.
Fully socketed — Gold bus connectors.
Assembled, Burned in, and Tested at 2MHz.

Deluxe Version with RAM Character Generator **$458.76**
Without RAM Character Generator **$398.74**
Also Available...
64 or 32 x 16 Video Board ... **$198.71**



THE CLASSY CHASSIS . $798.19
With Baud Rate Generator on Mother Board **$828.19**

32K SYSTEM *Incomparable Features... at a Comparable Price!* **$1,594.59**
Includes: Chassis, 6800 CPU, 32K RAM Board, Choice of I/O Card.

16K Version of above **$1,374.49**

- Ferro-Resonant Power Supply (+8V at 25 Amps, + and − 16V at 5 Amps each.)
- 6800/6809 Mother Board, has fifteen 50 pin plus 8 DIP-switch addressable 30 pin slots, fully decoded to 4, 8 or 16 addresses — Gold Plated Pins.
- Heavy Weight aluminum cabinet with fan and provisions for 1 or 2, 5 inch disk drives.

Phone, write, or see yourdealer for detailsand prices on our broad range of Boards and Systems for the SS50/SS50C bus and our AC Power Control Products for all computers.

GIMIX INC.
1337 WEST 37th PLACE, CHICAGO, IL 60609
(312) 927-5510 • TWX 910-221-4055

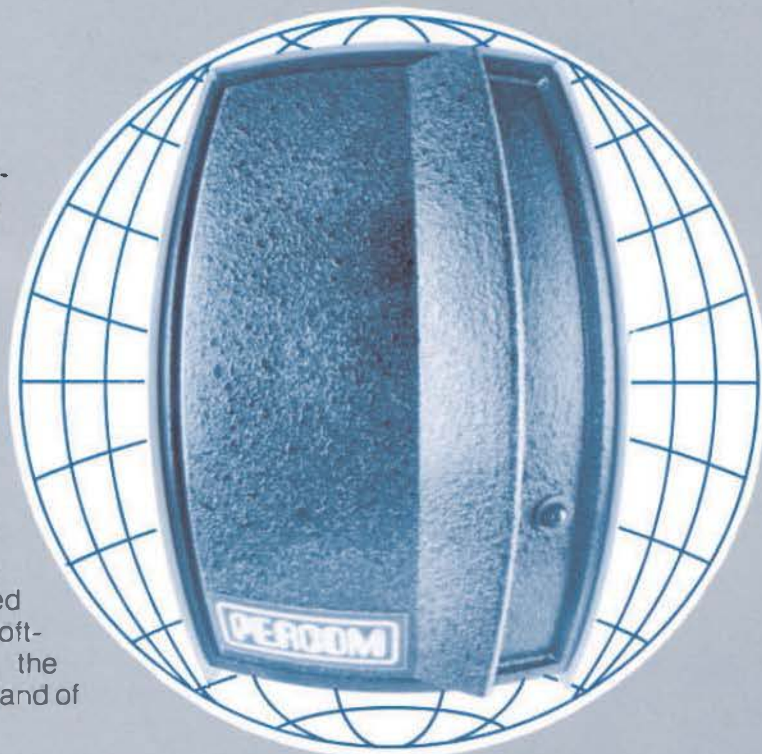
The Company that delivers.
Quality Electronic products since 1975.

GIMIX® and GHOST® are Registered Trademarks of GIMIX INC.

SEE GHOST AD PAGES 38 & 43

In the world of 6800 Microcomputing there is only one Universal Mini-Disk System . . .

the PERCOM LFD-400™ with SOFTRAN™



Made possible by SOFTRAN™, an innovative $24.95 translator program, the reliable Percom LFD-400™ has just been upgraded to the first universal mini-disk storage system.

Suddenly the two worlds of 6800 minidiskette software become one. Because the LFD-400™ with SOFTRAN™ can read either soft-sectored or hard-sectored disks.

And owning an LFD-400/SOFTRAN system means you can run minidiskette programs from the enormous combined selection of all of the principal 6800 software houses — TSC, Computerware, the Software Works, Hemenway Associates and of course Percom.

Available in versions for mini FLEX†, FLEX 2.0† and Smoke Signal Broadcasting Company's DOS, SOFTRAN™ copies soft-sectored minidiskettes track-for-track onto hard-sectored minidiskettes. If the source disk includes a FLEX† or 'Smoke' DOS, SOFTRAN™ is used to modify the operating system to function with the Percom LFD-400™.

SOFTRAN™ is supplied on a minidiskette along with utilities for only $24.95. A users manual is included. You must indicate whether SOFTRAN™ is to be used for mini FLEX†, FLEX 2.0† or Smoke's DOS.

The Percom LFD-400™ mini-disk system sells for only $599.95, complete with: (1) the drive, drive electronics and Percom's rugged PS-401 power supply all in a finished enclosure, (2) a demonstrably superior controller PC card featuring an explicit data/ clock separation circuit, MPX, a remarkable 2K DOS, and provision for 1K extra PROM, (3) an interconnecting cable and (4) a 70-page users manual.

Also available: Upgrade kits for SWTP or 'Smoke' mini-disk drive systems. Kit includes LFD-400™ controller, MPX DOS & SOFTRAN™. Only $224.95.

Available soon!
SOFTRAN™ for Percom's 77-track LFD-800™ mini-disk system; SOFTRAN/9™ for 6809 FLEX† files and programs.

™ trademark of Percom Data Company, Inc.

† trademark of Technical Systems Consultants, Inc.

PERCOM

PERCOM DATA COMPANY, INC.
211 N. KIRBY GARLAND, TEXAS 75042
(214) 272-3421

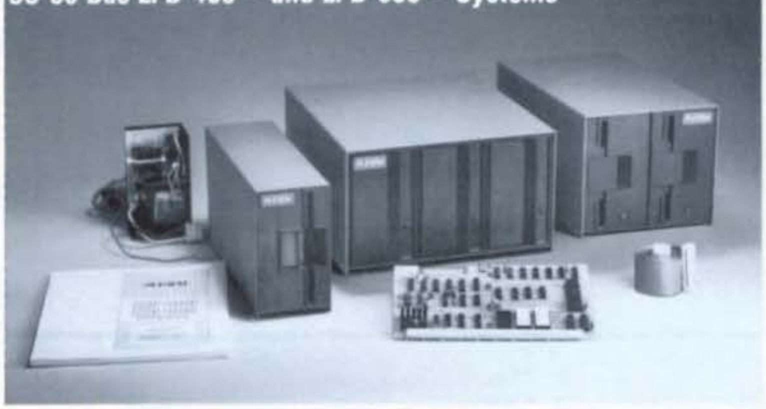
Percom *'peripherals for personal computing'*

To place an order or request additional literature call toll-free 1-800-527-1592. For technical information call (214) 272-3421. Orders may be paid by check, money order, COD or charged to a VISA or Master Charge account. Texas residents must add 5% sales tax.

PRICES AND SPECIFICATIONS SUBJECT TO CHANGE WITHOUT NOTICE

Welcome to Percom's Wide World

SS-50 Bus LFD-400™ and LFD-800™ Systems



Each LFD mini-disk storage system includes:

- drives with integral power supplies in an enamel-finished enclosure
- a controller/interface with ROM operating system plus extra ROM capacity
- an interconnecting cable
- a comprehensive 80-page users manual

Low-Cost Mini-Disk Storage in the Size You Want.

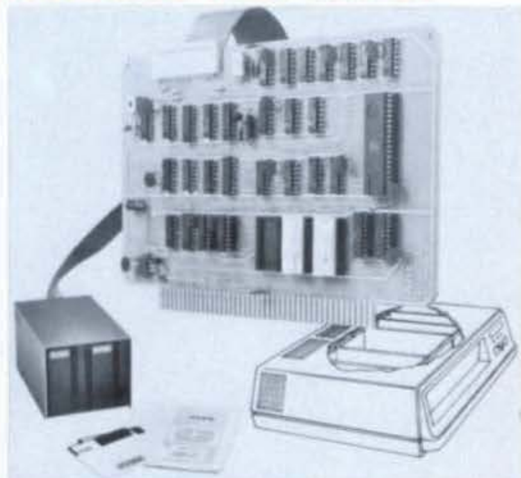
Percom LFD mini-disk drive systems are supplied complete and ready to plug in the moment they arrive. You don't even have to buy extra memory. Moreover, software support ranges from assembly language program development aids to high-speed disk operating systems and business application programs.

The LFD-400™ and -400EX™ systems and the LFD-800™ and -800EX™ systems are available in 1-, 2- and 3-drive configurations. The -400, -400EX drives store 102K bytes of formatted data on 40-track disks, and data may be stored on either surface of a disk. The -800, -800EX drives store 200K bytes of formatted data on 77-track disks.

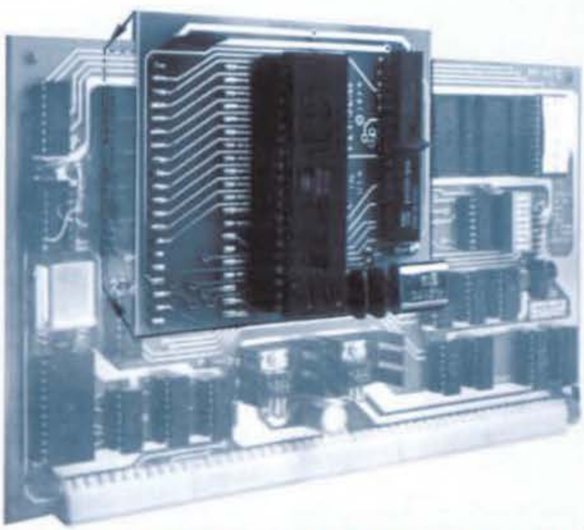
The LFD-1000™ systems (not pictured) have dual-drive units which store 800K bytes on-line. The LFD-1000™ controller accommodates two drive systems so that a user may have as much as 1.6M bytes on-line.

Mini-disk storage system prices:

MODEL	1-DRIVE SYSTEM	2-DRIVE SYSTEM	3-DRIVE SYSTEM
For the SS-50 Bus:			
LFD-400™	$ 599.95	$ 999.95	$1399.95
LFD-800™	895.95	1549.95	2195.95
For the EXORciser* Bus:			
LFD-400EX™	$ 649.95	$1049.95	$1449.95
LFD-800EX™	945.95	1599.95	2245.95
LFD-1000™	(dual) $2495.00	(quad) $4950.00	—



EXORciser Bus LFD-400EX,™ -800EX™ Systems*



Upgrade to 6809 Computing Power. Only $69.95

Although designed with the SWTP 6800 owner in mind, this upgrade adapter may also be used with most other 6800 and 6802 MPUs. The adapter is supplied assembled and tested, and includes the 6809 IC, a crystal, other essential components and user instructions. Restore your original system by merely unplugging the adapter and a wire-jumpered DIP header, and re-inserting the original components. Also available for your upgraded system is PSYMON™ (Percom SYstem MONitor), the operating system for the Percom 6809 single-board computer. PSYMON™ on 2716 ROM costs only $69.95. On diskette (source and object files), only $29.95.

Data Terminal & Two-Cassette Interface — the CIS-30+

- Interface to data terminal and two cassette recorders with a unit only 1/10 the size of SWTP's AC-30.
- Select 30, 60 or 120 bytes per second cassette interfacing; 300, 600 or 1200 baud data terminal interfacing.
- Optional mod kits make CIS-30+ work with any microcomputer. (For MITS 680b, ask for Tech Memo TM-CIS-30+-09.)
- KC Standard/Bi-Phase-M (double frequency) cassette data encoding. Dependable self-clocking operation.
- Ordinary functions may be accomplished with 6800 Mikbug* monitor.

Prices: Kit, $79.95. Assembled, $99.95. Prices include a comprehensive instruction manual. Also available: Test Cassette, Remote Control Kit (for program control of recorders), IC Socket Kit, MITS 680b mod documentation and Universal Adapter Kit (converts CIS-30+ for use with any computer).

of 6800 Microcomputing.

6800/6809 SOFTWARE

System Software

6800 Symbolic Assembler — Specify assembly options at time of assembly with this symbolic assembler. Source listing on diskette $29.95

Super BASIC — a 12K extended random access disk BASIC for the 6800 and 6809. Supports 44 commands and 31 functions. Interprets programs written in both SWTP 8K BASIC (versions 2.0, 2.2 & 2.3) and Super BASIC. Features: 9-digit BCD arithmetic, Print Using and Linput commands, and much more. Price $49.95

TOUCHUP™ — Modifies TSC's Text Editor and Text Processor for Percom mini-disk drive operation. Supplied on diskette complete with source listing $17.95

Operating Systems

INDEX™ — This easy-to-use disk-operating and file management system for 6800 microcomputers is fast. I/O devices are serviced by interrupt request. INDEX™ accesses peripherals the same as disk files — new devices may be added without changing the operating system. Other features: unlimited number of DOS commands may be added • over 60 system entry points • display only those files at or above user-specified file activity level • versions available for SWTP MF-68, Smoke's BFD-68 and Motorola's EXORciser*. Price $99.95

MINIDOS-PLUSX™ — An extension of the original MINIDOS™ for LFD-400™ mini-disk systems, MINIDOS-PLUSX™ manipulates files by six-character names. Supports up to 31 files. Resident commands include Initialize, Save, Allocate, Load, Files (directory list), Rename and Delete. Supplied on 2708 ROM with a minidisk file that includes transient utilities such as Copy, Backup, Create, Pack and Print Directory. Price $34.95

PSYMON™ — Percom SYstem MONitor for the Percom single-board/SS-50-bus-compatible 6809 computer accommodates user's application programs with any mix of peripherals **without** modifying programs. PSYMON™ also features character echoing to devices other than the communicating device, sophisticated register and memory dump routines and more. Price (on 2716 ROM) $69.95

WINDEX™ — Described in detail elsewhere on this page.

Business Programs

General Ledger — For 6800/6809 computers using Percom LFD mini-disk storage systems. Requires little or no knowledge of bookkeeping because the operator is prompted with non-technical questions during data entry. General Ledger updates account balances immediately — in real time, and will print financial statements immediately after journal entries. User selects and assigns own account numbers; tailors financial statements to firm's particular needs. Provides audit trail. Runs under Percom Super BASIC. Requires 24K bytes of RAM. Supplied on minidiskette with a comprehensive users manual. Price $199.95

FINDER™ — This general purpose data base manager is written in Percom Super BASIC. Works with 6800/6809 computers using Percom LFD-400™ mini-disk drive storage systems. FINDER™ allows user to define and access records using his own terminology — customize file structures to specific needs. Basic commands are New, Change, Delete, Find and Pack. Add up to three user-defined commands. FINDER plus Super BASIC require 24K bytes of RAM. Supplied on minidiskette with a users manual. Price $99.95

Mailing List Processor — Powerful search, sort, create and update capability plus ability to store 700 addresses per minidiskette make this list processor efficient and easy to use. Runs under Percom Super BASIC. Requires 24K bytes of RAM. Supplied on minidiskette with a users manual. Price $99.95

From the Software Works

Development and debugging programs for 6800 μCs on diskette:

Disassembler/Source Generator	$30.95
Reloc'ing Disas'mblr/Segmented Text Gen	$40.95
Disassembler/Trace	$25.95
Support Relocator Program	$25.95
Relocating Assembler/Linking Loader	$55.95
SmithBUG** (2716 EPROM)	$70.00

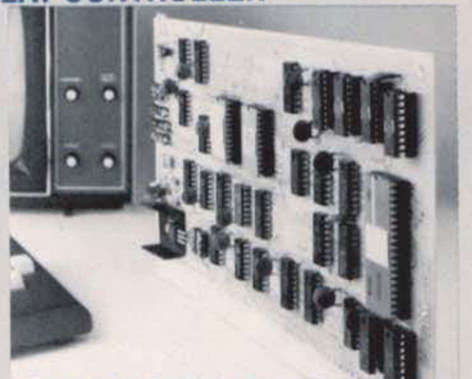
½-Price Special on Hemenway Software!

CP/68‡ disk operating system	$ 49.97
STRUBAL+‡ compiler	$124.97
EDIT68 text editor	$ 19.97
MACRO-Relocating Assembler	$ 39.97
Linkage Editor (LNKEDT68)	$ 24.97
Cross Reference utility	$ 14.97

™trademark of Percom Data Company, Inc.
* trademark of Motorola Corporation
‡Trademark of Hemenway Associates Company
**SmithBUG is a trademark of the Software Works Company

This programmable **VIDEO DISPLAY CONTROLLER** processes display changes instantly in real-time. The Electric Window™ resides completely in main memory so control is accomplished by direct MPU access to the character-store memory and display control registers. Peer at the screen and you look right into video display memory space while you input and manipulate text — an indispensable feature for efficient screen editing and word processing. The Electric Window™. It's worth looking into. Features include:

- **Programmable CRT controller chip** that provides extraordinary versatility in software control of horizontal and vertical formatting, cursor positioning, scrolling and Start/Reset functions.
- A standard ASCII 128-unit ROM character generator which generates easy-to-read **7 x 12 dot-matrix characters with lower case descenders**. Plus . . .
- Provision for an **optional ROM** that may be programmed **for special symbols or characters**.
- Resides entirely in **2K on-board RAM** mapped into main memory.



The Electric Window.™ Worth Looking Into. $249.95

- An optional **software driver program** called WINDEX™ that complements the fast, hardware-implemented functional capability of the controller. WINDEX™ will auto-link to PSYMON™, the monitor for the Percom SBC/9™ single board computer. The ROM version of WINDEX™ costs $39.95. The minidiskette version (with source and object files) sells for $29.95.
- **Up to 24 80-character lines** — programmable.
- Program control of **display highlighting**.
- Program **interlaced or non-interlaced** scan.
- Use either **standard video monitor or modified tv**.

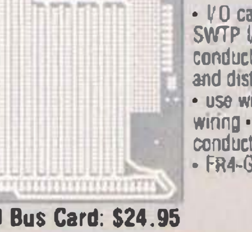
Now Available! the SBC/9™ MPU/Control Computer

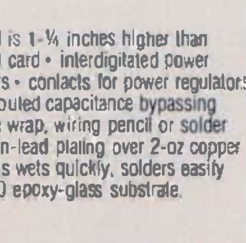
(Single-Board-Computer/6809) — stands alone as a control computer, but also compatible with the SS-50 bus for use as an MPU card. Includes PSYMON™ (Percom SYstem MONitor) in a 1K ROM and provides for additional 1K of ROM. Also includes 1K of RAM. Features: Super Port — provision for multi-address, 8-bit bidirectional data lines • an intelligent data bus for multi-level data bus decoding • an on-board 110-baud to 19.2 kbaud clock generator • extended address capability — to 16 megabytes — without disabling baud clock or adding hardware. And much more. Supplied with PSYMON™ and comprehensive users manual. Price .... $199.95.

See full page ad elsewhere in this magazine for all of the SBC/9™ features.

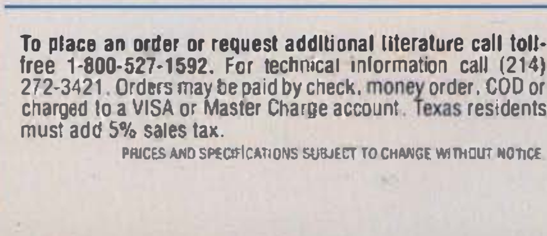
Full Feature Prototyping PC Boards

All of the features needed for rapid, straightforward circuit prototyping. Use 14-, 16-, 24- and 40-pin DIP sockets • SS-50 bus card accommodates 34- and 50-pin ribbon connectors on top edge, 10-pin Molex connector on side edge • I/O card accommodates 34-pin ribbon connector and 12-pin Molex on top edge





I/O Bus Card: $14.95



• I/O card is 1-¼ inches higher than SWTP I/O card • interdigitated power conductors • contacts for power regulators and distributed capacitance bypassing • use wire wrap, wiring pencil or solder wiring • tin-lead plating over 2-oz copper conductors wets quickly, solders easily • FR4-G10 epoxy-glass substrate.

SS-50 Bus Card: $24.95

To place an order or request additional literature call toll-free 1-800-527-1592. For technical information call (214) 272-3421. Orders may be paid by check, money order, COD or charged to a VISA or Master Charge account. Texas residents must add 5% sales tax.

PRICES AND SPECIFICATIONS SUBJECT TO CHANGE WITHOUT NOTICE.

PERCOM

PERCOM DATA COMPANY, INC.
211 N. KIRBY GARLAND, TEXAS 75042
(214) 272-3421

68 MICRO JOURNAL SOFTWARE CONTEST

Prize list now at $15,000.00+ and growing.

Prizes for each category will be:

FIRST - Life Subscription 68 Micro Journal™

SECOND - 6 year extension 68 Micro Journal™

THIRD - 3 year extension 68 Micro Journal™

4th-10th 1 year extension 68 Micro Journal™

The software must be applications, utilities or serious software, of original design, to operate with the following CATEGORIES:

TSC FLEX	6800 Disk System MiniFlex
TSC FLEX	6800 Disk System FLEX Ver. 2.0 5"
TSC FLEX	6800 Disk System Ver. 1.0 8"
TSC FLEX	6809 Disk System Ver. 09 5" or 8"
SSB	Dos Version 5
SSB	Dos Version 4 or earlier
SSB	TSC FLEX Version
PERCOM	INDEX
PERCOM	MiniDos+
HEMENWAY	CP/68 Disk System
MSI	Disk Operating Systems
SOFTWARE DYNAMICS SDOS	
JPC	TC-3 Cassette System
ANY	KC Standard Tape System
BASIC	Any 6800 Version
BASIC	Any 6809 Version

There are sixteen (16) categories, as indicated above. In addition we have other prizes donated by various vendors of 6800/09 products. As of the 15th of November 1979 over $13,000.00 in prizes has been pledged.

GIMIX - Mainframe, value $829.00. ** A GRAND PRIZE **

SOUTHWEST TECHNICAL PRODUCTS CORP (SWTPC) Computer 69/8, value $595.00. ** A GRAND PRIZE **

DIGITAL RESEARCH: COMPUTERS, 16K static memory board kit, value $295.00.

JPC PRODUCTS, TC-3 kit w/CFM-3 package; AD-16 kit; CK7 kit, total value $190.00.

STAR-KITS, Set checkbook balancing software : MiniFLEX™ or PERCOM Super BASIC on disk, value $40.00.

MICROWARE Systems Corp, Package 1 each: ABASIC Compiler, ABASIC Source Gen., ABASIC interpreter, LISP interpreter; also 6 each RT/68 ROM OS and 6 each 6800 Chess programs. Total value $1,005.00.

SMOKE SIGNAL BROADCASTING, two $500.00 gift certificates for any SSB product. Categories eligible SSB DOS Ver. 4 or earlier and Ver. 5 or later; total value $1,000.00.

LUCIDATA, 5 each PASCAL Ver. 2, 1 to each FLEX™ category. Total value $750.00.

COMPUTERWARE, 2 each $200.00 gift certificates for any Computerware software product. 1 gift certificate for best of SSB disk BASIC and 1 gift certificate for best of Computerware's Random BASIC. Total value, $400.00.

SSI, Schreier Software Index, choice of $100.00 of SSI software.

SOFTWARE DYNAMICS, 6800 BASIC Compiler, value $350.00.

The MICRO WORKS, choice of any one item, value up to $179.95.

HEMENWAY ASSOCIATES, INC., Books, 3 each CP/68, 3 each XA6809 Macro Cross Assembler, 3 each STRUBAL+ Compiler, total value $329.55.

CER-COMP Microcomputers, MiniDisk+ Disk system (EPROM) and software on disk. Total value $89.00.

TECHNICAL SYSTEMS CONSULTANTS (TSC), $250.00 choice of any TSC software, for best of FLEX™ entries.

HHH Enterprises and Springbok Digitronics, SPIRIT (disk SSB) and a copy of STD-1, these will be awarded for 'best of SSB DOS. Value $110.00.

ED SMITH'S SOFTWARE WORKS, RRMAC Recursive Macroassembler and Linking Loader, $150.00, for best recursive macro program.

PERCOM, Assorted hardware and software, items to be listed next month

Final decision shall be delegated to a panel of judges selected by the staff of 68 Micro Journal™. All judges decisions are final and each person submitting, shall by his or her submitting material for evaluation, acknowledge that they agree to abide by any and all rules of this contest, as published within the pages of 68 Micro Journal™

Programs and material submitted shall be judged on the basis of good and workable software. By this we mean, it should do something useful and be needed by the average 6800/09 user in the particular category. Size is of little importance, the most important consideration will be how useful it is.

All material submitted shall remain the property of the original owner (who should be the author). Each submission shall contain a paragraph that states the material submitted is of original design and the property of the person in whose name it is submitted.

It shall be understood that regardless of who wins or does not win a prize, all material submitted shall be authorized and eligible, to be published by 68 Micro Journal™. Material published, which was not a winning entry, shall gain the author an extension to his or her subscription. Anyone may enter and it is not a requirement that the person submitting material be a subscriber to 68 Micro Journal™. Prizes will be awarded on the quality of the material submitted and being or not being a subscriber, will have no bearing.

Full details may be secured from previous issues of 68 Micro Journal™.

AN I N D E X
to the
"68" MICRO JOURNAL
Compiled by Jim Schreier

Preface

Few Microcomputer publications can match the variety of the 1979 issues of the "68" MICRO JOURNAL. The 200 plus entries of this Index will be proof enough. My 6800 interest was created a few years ago when when it was discovered the 6800 systems were the only ones to make sense. And they worked. And worked. In almost three years my SWTPC 6800 went down once. It blew a fuse. So you see, the "68" MICRO JOURNAL has a good act to follow: It makes sense and provides excellent information.

The thousands of MICRO JOURNAL readers probably keep back issues under protective custody. And, based on the assumption that the average reader's ability to find a specific article, news release, product announcement, review or letter is no better than mine, an Index is a must. Some microcomputer magazines are made to look nice, some aim at the hippie market (yet!); but the MICRO JOURNAL is, like a three course meal, made to be enjoyed and digested.

This Index covers everything but ads. Some of the page layouts in early issues are not clear, however any Index errors are my responsibility. The Index was prepared using the TSC FLEX2 Text Editor, Text Processor and Sort/Merge Package. Five fields were established after examining the various type source entries. Since the sources contain two type of entries the fields had to have certain common elements for proper layout. Normally the "no entry" character ("-") would be edited out prior to the final Text Processor pass. In order for MICRO JOURNAL users unfamiliar with Text Processing to observe these items, "no entry" characters have been retained. The FLEX2 DOS TTYSET WD command was set at 42.

It is my hope that the labor represented in this Index may be of current and future value to that special group of people, the "68" MICRO JOURNAL readers.

6800 Software FLEXtm DOS commands from EDITOR (assembled listing with comments) By Art Weller November/December 1979 p 27
6800 Software Data acquistion network (assembled listing with some comments) By Don Aldridge November/December 1979 p 32
6800 Software Simple Line Editor (assembled listing with comments and symbol table) By Chuck Adams October 1979 p 26
6800 Software FLEXtm pause control routine (assembled listing without comments) By James Caldwell October 1979 p 30
6800 Software Procedure library (assembled listing with comments) By Gary Magnusen October 1979 p 33
6800 Software SWTPC BASIC to FLEXtm DOS Interface Command (assembled listing with comments) By - August 1979 p 17
6800 Software Transfer (FLEX1tm to FLEX2tm) assembled listing with comments and symbol table By - May 1979 p 18
6800 Software BOOT (FLEXtm - BFD) assembled listing with comments and symbol table By - May 1979 p 33
6800 Software Paper Tape Reader Program (assembled listing with comments and symbol table) By - May 1979 p 38
6800 Software Diskedit (assembled listing without comments; symbol table and object code) By R. P. Lajeunesse Sept 1979 p 30
6800 Software miniFLEXtm disassemble source code generation (assembled listing with comments) By Robert Boyd September 1979 p 10
6800 Software SSB DOS Backspace (assembled listing with comments) By Dan Johnson September 1979 p 8
6809 Documentated indirect vector addresses - By - September 1979 p 15
6809 Emulator by the Micro Works article By Ted Feintuch May 1979 p 23
6809 for the 6800 product article By Byron Seastrunk August 1979 p 5
6809 Patches to NEWDISK command - By John Byrns October 1979 p 19
6809 question letter By Darin Adler November/December 1979 p 23
6809 Software PYSOM Monitor (assembled listing with symbol table) By PERCOM November/December 1979 p 17
6809 Software LIFE (assembled listing with comments) By Cliff Rushing September 1979 p 28
6809 SS-50 Bus Alterations and Bus Line Description - By - September 1979 p 14
6809 Undocumentated indirect vector addresses - By - September 1979 p 15
6809--programming article By Terry "Tex" Ritter November/Decmeber 1979 p 34
6809--PYSOM Monitor program listing By PERCOM November/December 1979 p 17
6809--PYSOM Monitor product description By PERCOM October 1979 p 10
A 6800 Timing Delay article By Howard Berenbon May 1979 p 21
A Look at the SWTPC CT-82 Video Terminal product review By Mickey E. Ferguson March/April 1979 p 7
A Review of Some 6800 Monitors article By Robert C. Boyd June 1979 p 32
AAA Chicago Computer Mart Editor Modifications to GIMIX-BUG news release By - July 1979 p 16
Advertising and objectives editorial By D. Williams May 1979 p 8
Advertising honesty editorial By D. M. Williams June 1979 p 40
Algebraic to Polish Expression Translation Crunchers Corner By Jack Bryant November/December 1979 p 8
An Integer Arithmetic Package Crunchers Corner By Jack Bryant May 1979 p 9
Atlanta Hamfest pictures By staff August 1979 p 20
Back Issue Information - By - September 1979 p 12
Back issue policies editorial By D. M. Williams May 1979 p 17
Basic Cassette Formats article By Phil Schuman June 1979 p 41
BASIC Renumbering for SWTPC 8K BASIC article By Mickey Ferguson June 1979 p 17
BASIC Software TSC FLEX2 BASIC Pause By - September 1979 p 26
BASIC Software Lower Case to Upper Case By Geoffrey A. Gass September 1979 p 29
BASIC Software--Computerware BASIC Some Common BASIC Games (MAZE disk creation & EXPLORE) By Tom Harmon May 1979 p 27 & 28
BASIC Software--Computerware BASIC Christmas Card List By Art Weller November/December 1979 p 27
BASIC Software--Computerware v 7.0 & 8.0 Mail List By Gene Embry September 1979 p 32
BASIC Software--Computerware v 7.0 and 8.0 Data Files By Byron Seastrunk August 1979 p 8
BASIC Software--General BASIC JUMBLE word game By E. M. Pass July 1979 p 34
BASIC Software--General BASIC Magic Squares By Paul Pennington June 1979 p 39

Micro-time 6800 Review

The Micro-time 6800 is a stand alone real-time clock and calendar. This means that no system overhead or interrupts are required for time keeping as in some other system clocks. The clock is timed by a quartz oscillator with a trimmer capacitor for fine tuning. This allows operation without need for timing from the sixty Hertz power source and steering diodes are provided to allow operation from separate dc power sources or for battery backup to keep the clock running during power blinks.

The clock board plugs into a standard 30 pin I/O port and has a connector on top where manual time and date set switches may be attached.

The software provided with the board is in three sections. The first section when called as a subroutine, updates the time and date in a scratch area in ram. The second routine prints the time and date on the I/O device in the form : "12/17/79 10:35:22 PM EST". The last routine is used to set the clock and calendar.

The clock board is offered just as a bare board with connectors and documentation or factory assembled and tested. The assembled unit is available in either a switch setable version or software setable. The version which was sent for evaluation was the software setable version and seems to be well worth the few extra bucks that it costs.

The software was provided with a commented source listing, which was fortunate since I had to reassemble for my 09 system. Incidentally, when reassembled for the 09, the code was about 101 longer; however, when optimized for the 09, the needed code required approximately 151 fewer bytes than the 6800 version.

The documentation supplied is complete and explains operation adequately. The quality of the board and components used is good and the advantage of not having to worry about loosing the time every time the reset button is pressed, as happens with the clock I have been using, makes this device a worthwhile addition to any 68XX system. Other accessories include an A/C adapter for powering the clock while the computer is turned off and a Kansas City cassette with the previously described software. The Micro-time 6800 is available from :

THE DATA MART
914 E. WAVERLY DRIVE
ARLINGTON HEIGHTS, IL 60004

A 68 Micro Journal™ lab rating of: AAA

Rating Scale:
AAA - Excellent
AA - Good
A - Fair (could be better but works)
P - Poor (may not always work properly)
X - Not recommended for children
(or anything else!)

PERCOM PROTOTYPING BOARDS REVIEW

Percom Data Company has recently announced two new 68XX prototyping circuit cards. One card fits the standard SS-50 bus and has enough room to accomodate up to 70 14-pin DIP sockets or less 16,24, or 40-pin sockets. The top of the board has pads for insertion of a 34-pin and a 50-pin ribbon cable connector. One side of the board provides an area for miscellaneous circuitry or test points. DC power for the board is fed from the 8-volt bulk supply and goes to circiut pads for a 5-volt regulator. From the regulator a plus supply buss feeds between alternate rows of pads so that it is close to all circuits. A ground buss passes between opposite alternate rows also in close proximity to all circuits.

The second board is a SS-30 bus board. It has room for up to 34 14-pin DIP sockets or less larger sockets.The top edge provides pads for one 12-pin Molex connector and a 34-pin ribbon cable connector. A small area is also provided at the top for other miscellaneous circuits,indicators, or test points. The SS-30 board also has pads for a 5-volt regulator and positive supply and ground busses close to all socket locations. Both boards are single sided, solder plated, and up to Percoms usual high quality.

One thing that I watch closely on 68XX boards is how well the mother board molex connectors fit. There is nothing that bothers me more than boards that fit onto the mother board at a 30-degree angle. The Percom boards passed this test well. The quality of the boards is excellent and allow for optimun placement of IC's and components. The only complaint that I have is the fact that there is no identification of any of the mother board signals on the board. This would have made use of the boards somewhat easier. The boards are available from:

PERCOM DATA COMPANY, INC.
211 N. KIRBY
GARLAND, Texas 75042

A 68 Micro Journal™ lab rating of: AAA

Rating Scale:
AAA - Excellent
AA - Good
A - Fair (could be better but works)
P - Poor (may not always work properly
X - Not recommended for children
(or anything else!)

J.B.I. CT1024-64 High Speed Conversion

If you are like a lot of the rest of us 68XX users, you still use one of the SWTPC earlier terminals. The SWTPC CT-1024 and CT64 were two of the most popular video display units for 68XX users. They were low in cost and worked reasonably well. The keys were sometimes balky or at other times self striking. This was annoying but not fatal to the operation. The screen when refreshed looked like a midwest blizzard and occasionally the cursor developed a mind of it's own,

wandering here and there at will. One of the most annoying drawbacks was the slow write speed of either. Three hundred baud was the normal and twelve hundred baud was the upper limit, and still is until you install the J.B.I. conversion kit. Despite these and other occasional quirks; the CT-1024 and CT-64 were and still are in use by thousands of micro users worldwide today.

The J.B.I. conversion kit eliminates many of the major drawbacks of the CT series of video displays. It can be adapted to those units that have been field-updated with 64 character mod (CT-1024) and other popular changes. It uses a DMA method of screen memory, essentially causing the terminal memory to become computer memory. By this scheme the screen can be written to at near computer speed. Screen writes can range from one character per second to 4,000 characters per second (40K baud). All of the memory management (terminal) is still accomplished by the terminal and leaves the CPU unburdened for these chores. BASIC can 'POKE' directly any character position (limited grapics) and 'PEEK' any character position.

Terminal memory can be relocated to any 1K block in computer memory range that is available. This requires a software change of three standard Mikbug™ routines. These are the ones used in screen write, e.g. $E1D1 OUTEEE, $E1AC INEEE and $E07E PDATA. Patches are furnished for practically all popular software.

The board has been run on 2 meg machines and requires no delay. Included with the kit is a scource listing of all changes or patches. The supplied software is RDMable. The converted terminal allows software control over scrolling or paging. One foil cut on the terminal eliminates the 'snow' problem when using the conversion kit. Baud rates are controlled from the keyboard or from software. It honors the tape 'SAVE and LOAD' thru BASIC at 300 baud. Exisiting software reqires only a change of the three routine references to run in the converted mode.

COMPUTER MODS

If you are still using the MP-C I/O board in slot 1 you will be required to lift one IC pin on the board, this eliminates 'echo' to the terminal. No changes are required to the computer if you are using a serial 'MP-S' I/O board.

MODS FOR CT-64

One trace cut and two IC pin cuts (or lift out if you are using sockets) also one wire jumper added.

MODS FOR CT-1024

One IC pin lift or cut and two grounds extended. One or possibly two wire extensions.

The conversion comes with two boards connected by two ribbon cables. The boards are factory built and require only memory chip installing if you use your own. One board fits on the computer S50 bus and the other replaces the memory board in the terminal. The kit comes with or without out memory IC's. This way you can use your old 2102 memory chips (if they are in sockets) or can be ordered with all new memory chips (2 Mhz). The prices advertised are $169.00 with you supplying the memory chips. If you order with new memory chips the price is $179.00. We recommend that you order with new chips as most all older chips are slow and end up looking like worms are eating portions of some characters, interesting but annoying!

One note of caution if you are going to update a CT-1024 you need to let them know if it has been modified for 64 characters per line or is original.

The documentation seems very complete and should be useable by anyone who originall constructed his terminal. It comes with 12 pages of instructions, diagrams, board layouts, software patches and assembled source code.

Additional information can be secured from:

JOHNSON MICRO COMPUTER
2607 E Charleston
Las Vegas, Nevada 89104
1-702-384-3354

A 68 Micro Journal™ lab rating of: AAA

Rating Scale:
AAA - Excellent
AA - Good
A - Fair (could be better but works)

P - Poor (may not always work properly)
X - Not recommended for children
(or anything else!)

CORESIDENT JBUG AND MINIBUG II MONITOR ROM FOR MEK6800S2 MICROPROCESSING SYSTEM

K. Russell Peterman Staff Scientist
Radian Corp. 8500 Shoal Creek, Austin, TX 78766

The Motorola MEK6800D2 microprocessor system features a hexadecimal keypad for data/address entry and a 7-segment LED array for data/address display. The system also utilizes one ACIA as a Kansas City Standard audio cassette interface. The JBUG ROM monitor supplied with the system will support the functions outlined in Table I, including audio cassette read/write capabilities. However, during applications program development it is much more productive to use an external crossassembler such as M68SAM (Ref.1) for building object files. When the crossassembler is resident in a larger computer or development system, this almost always implies an RS-232 serial interface standard for data communications between the data terminal and the resident crossassembler. Thus, for prototyping purposes, it would be ideal in a system such as the MEK6800D2 to provide coresiding ROM monitors to format I/O data for either the hex keypad/audio cassett (JBUG) or an RS-232 serial data communications port (MINIBUG II) (Ref.2) as shown in Fig. 1. Although Motorola has released an excellent applications note (Ref.3) outlining the modifications required to allow coresiding ROM monitors in the MEK680002, their scheme is somewhat complex, so that it can provide program control of which ROM monitor is addressed. However, in many applications the monitor need simply be selected manually using a front panel control switch. In this manner applications program development could proceed from a source file in the development system machine to an ASCII coded object file on a digital cassette tape. The object file may then be transferred from cassette tape, via RS-232 serial interface, to the RAM resident in the MEK6800D2 system using the MINIBUG II monitor. The ROM control may then be switched to JBUG to allow complete system control of the MEK6800D2 from front panel keypads.

To implement this scheme, only one control signal need be switched between the two ROMs as shown in Fig. 2. The signal designated as $\overline{ROM}$ is output to the selected ROM by the JBUG/MBUG switch, shown in the figure. The upper portion of the switch also parallels the Tx clock and the Rx clock of the ACIA (U23), as shown in Fig. 1, when the MBUG ROM monitor is selected. Adequate space is provided on the MEK6800D2 board to add the second MC6830 ROM as shown in Fig. 1 as well as the RS-232 driver-receiver shown in Fig. 2. Note also that in order to ensure that no data are lost during monitor switching the MC6800 microprocessing unit should be held in the reset condition while the JBUG/MBUG switch is changed. The MEK6800D2 system provides baud rate logic for standard rates up to 9600 baud which may be selected at the output taps of counter U17 The JBUG ROM is normally supplied with the MEK6800D2 system; however the MINIBUG II ROM may be specified separately by asking for an MEX68MIN II preprogrammmed MC6830 read only memory.

REFERENCES

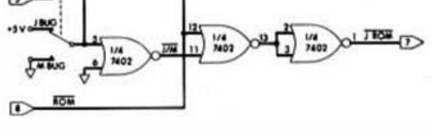
1. M68SAM is the property of Motorola SPD, Inc. Copyright 1974 to 1978 by Motorola, Inc.

2. Motorola Semiconductor Products, Inc., Applications Note AN-771, "MEK6800D2 Microcomputer Kit System Expansion Techniques", Motorola Semiconductor Products, Inc., Phoenix, Arizona, 1977.

3. Motorola Semiconductor Products, Inc., "Evaluation Module II User's Guide", Motorola Semiconductor Products, Inc., Phoenix, Arizona, 1976.

TABLE I

Monitor Function	JBUG	MINIBUG II
Display Registers	R	R
Load From Tape	L	L
Dump to Tape	P	P
Memory Examine/Change	M	M
Execute from Entered Address	G	G
Set Terminal Baud Rate	-	S
Test Memory	-	W
Punch Binary Tape	-	Y
Load Binary Tape	-	Z
Abort Program Execution	E	-
Trace (Single Step)	N	-
Set Breakpoint	V	-
Reset Breakpoint	V	-
Continue Execute from Breakpoint	E,G	-
Delete All Breakpoints	V	-



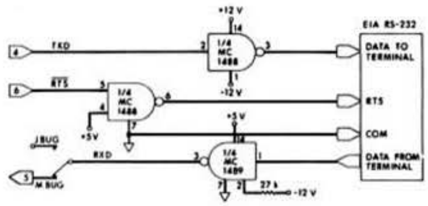


FIGURE 2
M6800 D II
DUAL MONITOR LOGIC
CIRCUIT DIAGRAM

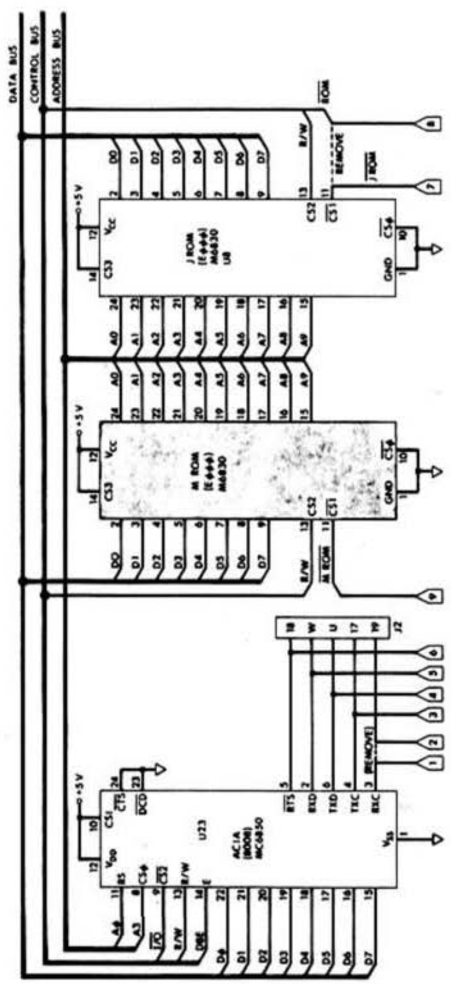


FIGURE 1
M6800 D II
DUAL MONITOR/I/O LOGIC
CIRCUIT DIAGRAM

***** CLASSIFIED ADVERTISING *****

USING THE 6801/6803 AND 6809 IN THE MP-A2 BOARD

Dr. J L Pentecost
3605 Clubwood Trail
Marietta, GA 30067

Both the 6801 (or 6803) and the 6809 MPU chips can be used with the SWTP MP-A2 board with simple adapters. This article describes this approach to use both the 6801/6803 and6809, in adapters, with TSC software.

First examination of the 6801/6803 specifications reveals a faster processor with equivalent 6800 instructions, plus some added instructions (Table I). Tests show that the 6801/03 runs typically 17.51 faster than the 6800 at the same clock frequency. The only disadvantage of the 6801/03 is the inability to use the first 20 Hex addresses in memory. These addresses have the special purpose registers and ports and are not available for memory use on the direct page. The pin-out of the 6801 and 6803 is not equivalent to the 6800 and an adapter is required. A circuit of an effective adapter is shown in Figure 1. With this adapter, and a jumper header substituted for the 6875 (as for the 6809, 68MJ6P6) on the MP-A2 board, the 6801 or 6803 runs most programs without any modification of the monitor or software. The only notable exception is disk versions of SWTPC BASIC (and possibly other versions). The TSC Editor, Assembler, BASIC, etc., operate properly.

Some additional advantages of the 6801/6803 include the availability of a programmable timer and a direct page ACIA port, the availability of 16 bit arithmetic shift instructions, PSHX, PULX and 8 bit multiply. The only disadvantages are the loss of 20 Hex bytes of memory on the direct page, and the inability to use DMA readily. Since most software for the 6800 also runs on the 6801/03 little difficulty should be experienced with this modification.

THE 6809

The 6809 may be used with the SWTPC MP-A2 board with simple adapters like the PERCOM. Some difficulty was experienced with this adapter on my system since a PIA port would not operate properly. The solution was found in pulling VMA high with a 1K resistor rather than by using E

AND Q. The 6809 in a simple adapter requires a new monitor. The easy approach is to modify the SWTPC S-BUG monitor for the standard 8000 Hex I/O normally used with the MP-A2 board. This is simpler than modifying the MP-A2 board to allow the EOXX addresses to be put on the main buss. These addresses are only used with the on-board EPROM or monitor. The contents of the monitor addresses in Table II should be changed from E0 to 80 to modify the ACIA location for the control port (only a MPS card!) and the 5 addresses for the mini- disk boot. This can be done by reading the standard S-BUG monitor into the EPROM programmer routine, modifying the addresses in memory, and programming a 2716 with the new code.

To modify TSC software to run on this system, it is only necessary to change the EOXX addresses in NEWDISK.CMD and FLEX.SYS. Once these addresses are changed, a new disk is formatted, these two programs copied onto the new disk and the modified FLEX.SYS LINKed, the new disk will boot and operate properly with the utilities. These changes may not be simple to make with only a 6809 disk and a single system. Here are two approaches.

First, if FLEX 2 is available (FLEX at 7000 will not work) it can be brought up and with memory from A000 to DFFF, GET,FLEX.SYS from the 6809 disk to place FLEX.SYS into memory. Change all E0 addresses (Table III) to 80 and SAVE,FLEX.NEW,C700,DF4D,CA00 on the 6809 disk. NEWDISK.CMD can be modified and saved similarly (see Table IV). This still does not allow the disk to boot even when linked, however, because the track 00 boot sector still contains E0 addresses. Only disks formatted with the modified NEWDISK command will boot with 8000 I/O.

If battery back-up for C000-DFFF is available to maintain FLEX 9 in memory, the system can be shut down, converted to a 6809 system and reset. Upon jumping to CD00 or CA00, FLEX 9 works properly. From here, NEWDISK a blank disk with the modified utility, copy FLEX.NEW to it and LINK,FLEX.NEW on the new disk.

This new disk will now boot properly and the system is up. The next approach can be used if FLEX 2 is not available and will work if no battery back-up is available for the RAM memory.

1. Boot the 6809 disk using the modified S-BUG and memory at C000-DFFF, reset, one sector will have loaded at C000.
2. Change all E0 addresses in this sector (Table V).
3. Set X to C000, jump to C000. This will cause the disk to load FLEX.SYS, but hang up, so reset again. Change all EOXX addresses (Table III) and jump to CD00.FLEX 9 will be operating.
4. SAVE,FLEX.NEW,C700,DF4D,CA00. Modify and save NEWDISK as above.
5. Format a new disk with the modified utility, copy FLEX.NEW and NEWDISK to this disk and LINK. The new disk will boot and all utilities will work properly.

Advantages:
1. No buss modifications or motherboard changes.
2. Low cost modifications allow use of both 6800 and 6809
3. All TSC software for 6809 can be used.
4. Up to 40K of memory is accomodated exactly as with the SWTPC board.
5. With the MOVE9 utility (by James Hughes), MINI-FLEX files are easily transferred to FLEX 9 disks.

Disadvantages:
1. Only 32K of useful memory is available vs 48K for I/O at EOXX.
2. Some initial software modification is required.

For those with Thomas Instrumentation video boards, a version of JOEBUG monitor (68MJ2) for the 6809 is also available to operate the video board, printer, keyboard and terminal ports simultaneously. FLEX 9 I/O must be modified for the video drivers however. It was noted in performing this modification that the jump table (D3E7-D3FC) is not normally used for CHAR in and CHAR out routines at 0370 and D3BB respectively and that jumps (7E XXXX) must be placed at D37D, INCH; D388, OUTCH; and D39C, STATUS to accomplish this modification.

TABLE I. NEW INSTRUCTIONS IN THE 6801/6803

ABX	B+X -> X
ADDD	M, (M+1) + D -> D
ASLD	C <- D <- 0
LDD	M, (M+1) -> D
LSRD	0 -> D -> C
MUL	A x 8 -> 0

```
PSHX        X -> Stack
PULX        Stack -> X
STD         D -> M, (M+1)
SUBD        D - M, (M-1) -> D
```

TABLE II. S-BUG MONITOR I/O ADDRESSES

F825 (ACIA)	FBC8	FBDD
FBB1	FBCD	FBEB
FBB4	FBD6	FBF0

TABLE III. FLEX 9 FLEX.SYS I/O ADDRESSES

D3E1(Timer)	DE79	DEB9
D3E3(ACIA)	DE88	DECB
D3E5(ACIA)	DE8B	DEE6
DE40	DE90	DEFC
DE48	DE98	DF23
DE58	DEB1	DF28
DE71		DF32

SAVE,FLEX.NEW,C700,DF4D,CA00

TABLE IV. FLEX 9 NEWDISK.CMD I/O ADDRESSES

C479	C4BE	C626
C47F	C5ED	C627
C48E	C5F6	C633
C499	C603	C63F
C49E	C618	C65E
C4A4	C624	

SAVE,NEWDISK8.CMD,C100,C6A7,CA00

TABLE V. I/O ADDRESSES IN BOOT SECTOR

C015	C04C	C067
C01E	C04F	C086
C02B	C054	
C040	C05B	

——————————————>>>

REFERENCE 'BOOKEEPING' NEXT COLUMN

NOTE: Due to the volume of data in the BASIC programs we will furnish copies of the entire disk programs (.BIN and .BAS) for $6.50 (miniFLEX® format) including postage and handling. The BASIC programs will be run next month in source format.

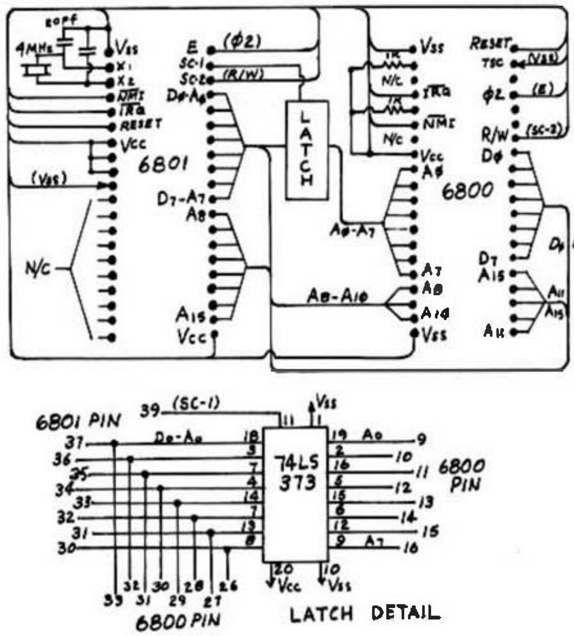


FIGURE 1 ADAPTER FOR USING 6801/03

BOOKEEPING (Disk & Tape)
MiniFLEX

William R. Stock
1125 Lois Dr
Cincinnati, OH 45237

Totally ignoring the fact that my father kept adequate financial records with nothing more than a check register and 39¢ worth of index cards, I have convinced all concerned that my SWTPC 6800 is useful because it keeps my books. Assuming you have similar problems, this bookkeeping system may be for you.

It is written in SWTPC BASIC 3.0, intended to be used with the Southwest minidisks under miniFLEX 1.0. If you have another system you may have to modify the programs a little.

HOUSEHOLD NEEDS

The primary purpose of a household bookkeeping system is to keep a record of all income and expenses; the former for your friend and mine, the IRS, and the latter for you. However, if we're going to do this on a computer we may as well go ahead and list our assets and liabilities (debts). This gives a General ledger, and a much more comprehensive picture of our financial status.

To this base I have added a rudimentary Accounts Payable, to assist in projecting cash requirements.

Notice that this system is designed for households. Businesses will still have to look elsewhere.

DESIGN PHILOSOPHY

In any monetary record keeping system, accuracy is of paramount importance. As a result, this system uses a double entry ledger. A double entry system can get unwieldy, however, if you have to keep track of assets, liabilities, debits

and credits, and how they interact. This leads to the second design consideration: simplicity.

be used if it is too complicated to operate. Consequently, this system was designed so that once it is booted up, all instructions are displayed on the CRT. The only debit/credit decisions you have to make are on the first entry of each transaction, and they are further simplified, as we shall show. Moreover, the programs call each other from disk, eliminating the need to remember what comes next.

SYSTEM REQUIREMENTS

This system was designed to run on (my) SWTPC 6800 with 20K RAM, a SWTPC 1024 terminal, SWTPC mini-floppies running TSC's miniFLEX 1.0 and BASIC 3.0 (the additional 4K required for miniFLEX is not included in the 20K), and the SWTPC AC-30 cassette interface + SWTBUG for tape backup of the data files.

The only programs unique to the SWTPC system are those dealing with the cassette. These programs are written in assembler, and interface with miniFLEX 1.0.

PROGRAM DESCRIPTIONS

POSTING: Transactions are entered into the books by the posting routine. The APIN program validates the transactions, makes the debit/credit decisions, and writes the transactions to the TRAN file. If any entries affect Accounts Payable, they are also written to the PTRAN file.

If any transactions were written to the PTRAN file, they are now sorted and the APMASTER is updated. If there are any rejected entries, they are displayed and the program aborted.

The Journal APPend program is called next, and it simply appends the transactions to those already present in the journal. This was done in BASIC to keep things simple.

Next, the TRAN file is sorted, and the GLMASTER is updated. At the same time, the sorted TRAN file is merged with GLHISTory, to produce an updated file of G/L transactions. Rejected entries are displayed on the CRT.

JOURNAL PRINT: Occasionally you will want to look at some transactions. This program lists them as they were input, within either a date or a sequence number range.

TRIAL BALANCE: This program lists the asset and liability accounts, with or without a listing of the transactions on file. It reads, but does not display, the income and expense accounts to arrive at the surplus, which is needed to make the debits equal the credits. These two figures will always match (unless you have posted to an account not on file.)

PROFIT & LOSS: You are sometimes interested in either how you made so much money, or where it all went. This program tells you. It reads the income and expense accounts, displays them, and by a clever algorithm (subtraction), gives you the amount of either profit or loss.

CASH REQUIRED: This is the only function of Accounts Payable at this time. The program tells you how much you need to meet the bills that will come due between the date of last update and the date you enter.

END OF PERIOD: Although the diskettes will probably hold a year's transactions, the update time can get rather long, so this program allows you to scratch the JOURNAL and the GLHISTory, with the option of saving the data on tape. Should you need to refer back to the saved data, the RECOVERY program will put the data back to disk, from which you can run the usual trial balance and journal print.

END OF YEAR: To set up your books for a new year, this program performs the end of period function, then zeros out the income and expense accounts and updates the net worth.

MAINTENANCE: You will have occasion to add and delete accounts. This routine performs the task. It also allows you to change an account name, and, on the accounts payable file, change the date and amount due.

BUDGET: Since everybody talks about budgets, I have included a budget program. It will tell you how much you've budgeted for each expense account so far this year, how much you've actually spent, and what percentage you've spent.

PART II: OPERATION

Once you have created all the disks, bringing up the system is a snap. Power up your computer, put the SYSTEM disk in drive #0, and enter D (SWTBUG). The computer will eventually respond with:

```
READY
#
```

to which you respond: CHAIN 0.START.

This routine gets today's date and saves it for all subsequent processing. You now select a job from the CRT menu.

1 - TRANSACTION POST. There are two rules to remember when posting transactions. First, ALL ACCOUNTS MUST BE ON FILE. Second, THE FIRST ENTRY DETERMINES DEBIT/CREDIT FOR THE ENTIRE TRANSACTION. We will now look at each of these in detail.

Since a typical household chart of accounts will contain between 75 and 100 entries, it is not feasible to verify account numbers on the input run. (It is possible, but response

time suffers terribly.) Consequently, it is imperative that you make sure each account number is correct. If a transaction contains items on file and items not on file, then some accounts get updated and some don't (because they're not there!), leading to an out of balance condition. (In which case you start over.)

My approach to the second rule might drive professional bookkeepers crazy, but it makes things simple. The computer knows, because you told it, which accounts are income, asset, liability, etc. It also knows, because the programs tell it, that increases to income are debits, and so forth. Furthermore, it can figure out, based on the first entry, the debit/credit status of each item. Consequently, to make things as simple as possible, I have adopted this rule: IF IT ADDS TO THE ACCOUNT BALANCE, ENTER IT POSITIVE. Consider the following transaction:

```
ACCT  AMOUNT  COMMENT
? 111 298.8 10/20

? 711 40.94
? 712 16.92
? 713 2.12
? 714 5.52
? 583 1.6
? 322 20
? 324 211.7
```

You will notice that the first entry is positive. It adds to 'Dad's income'. The computer knows that 111 is an income account, and makes a positive entry a debit. It also knows that accounts 711 through 583 (in the example) are expenses, and makes these entries credits. 322 and 324 are assets, and these will be debits. The important fact is that YOU don't have to worry about anything but the FIRST ENTRY. The rest is taken care of for you.

Less obvious, but equally important, the second through last entries add up to the amount of the first entry. This is the basis of the double entry system. You cannot post a transaction unless this condition is met, which means you cannot get out of balance (unless you post to an account that doesn't exist, as we've already mentioned.)

When you have made the last entry of a transaction, the 'in balance' condition triggers a neat, columnar display of the transaction. The program asks if everything is ok. At this point you should double check the account numbers, and then answer yes or no.

If you have entered the last line of a transaction and the program doesn't list it out, then the transaction obviously is not in balance. Figure out why, press return, and re-enter it.

When you are finished posting all your transactions, press return without entering anything. The program asks if you are REALLY finished. Answer yes (or no, if you're not!).

There is one last remark about the input program. If you look at the example, you will notice that it looks sloppy. This is not because I am a poor typist, but because I am lazy. It is far easier to hit the space bar than the comma, so the entire line is one INPUT A$ command. The program picks A$ apart, retrieving the account number, amount, and comment (if any). While this makes input easy, it does place one restriction on you: the fields must be separated by one, and only one, space.

2 - JOURNAL PRINT. The operator input to thee program is minimal. It asks if you want the entire journal printed. If you answer 'no', it asks if you want the range based on date or sequence number. Depending on your answer, it asks for the beginning and ending dates or sequence numbers.

3 - TRIAL BALANCE. Trial balance asks only one question: do you want detail to print. A 'yes' will display every transaction to every account that is on file. A 'no' will cause only the account number, description, and balance to print.

4 - END OF PERIOD. The only question is whether or not you want to save the data to tape. Answer yes or no.

5 - END OF YEAR. The same question for end of period is asked here.

6 - PROFIT & LOSS. There is no input to P&L. It runs all by itself.

7 - CASH REQUIRED. The cash required program must know the cutoff date you are interested in. It will add up the amounts due from the date of last update to the date you enter. Additionally, it asks whether or not you want it to display a list of which accounts and amounts are due.

The routine is limited by the fact that the amounts are added only once. Consequently, the date range should not encompass two payment periods.

8 - MAINTENANCE. Before we start talking about the input, let me remind you that all accounts must be accessed in ascending account number sequence. After you have all additions, deletions, and changes in sequence, you may start.

Enter the account number. If this is an old account you can 'return' through the description and it will stay the same. If it is a new account, enter the description. If it is an A/P account, enter the date due and amount due.

To delete an account, enter 'DELETE' for description. An account must have a zero balance to be deleted. Balances cannot be changed by maintenance.

To exit maintenance, hit 'return' without entering an account number.

10 - BUDGET. Printing a budget has no input and is no fun. Building a budget, however, is as close to a game as we will get with this pedestrian system.

First, let me warn you that the budget build program was designed to work off the previous year's actual expenses. As a result, it won't work until there is at least something in the general ledger balances.

The program starts out by asking you to select a budget period. Since we all pay as we go, we tend to think in weekly, monthly, etc., terms. And since our computer can multiply and divide, we will let it annualize our input.

Next, the program needs to know our best guess at our annual income. Guess too high and you'll be within budget, but show a loss. Guess too low and you'll have a rough time budgeting. I usually guess high, go over budget, and make New Year's resolutions.

The program now reads the general ledger expense accounts, computes what percentage of the total expense was spent on each account, computes what should be spent, and you're off to the races!

The repetitive display consists of the account name, the amount you have previously input, and the suggested amount. You now enter.

The item-by-item input is very flexible. It is so flexible, in fact, that it is simple to use and impossible to explain. Here are the possibilities:

(amount)(return): the amount entered replaces the previous amount. It is for the period chosen at the beginning of the program.

(amount),F (return): the amount entered replaces the previous amount as in the above example, and the account is then removed from further consideration (frozen).

(return): everything for this account remains the same.

F (return): Everything for this account remains the same and it is removed from consideration.

(amount),(period) (return): the amount entered is multiplied by the period entered, and the product divided by the initial period to arrive at the amount to be displayed. This is handy for things like insurance premiums.

(acct #),T (return): this places the entered account back into the matrix (thaws it). The disadvantage is that you have

When you have journeyed through all your expenses, the program will display how much you are over your income. It will then re-compute suggestions for all non-frozen accounts, and you play it again, Sam.

When you have finally figured out how to live within your income, the final budget will print, and you will be asked if you want to revise it. If you say 'no', the general ledger is updated with the new budget. (OBVIOUSLY you run the budget build before the end-of-year program!!!)

In closing, let me point out that you can't freeze nothing (zero amount). You can, however, freeze a penny, which will have minimal impact on the results.

PART III: CONVERSION

Building all the disks required is the most difficult aspect of the whole operation. Not only is everything unfamiliar, but you are dealing with a great volume of data, all of which must be entered correctly. Take heart; you have to do it only once.

Since miniFLEX file organization precludes a destructive update (rewrite), father/son is the only technique available. This means the masterfile must be in ascending account number sequence.

Since the programs assume the lowest account numbers are income accounts, followed by receivables, assets, expenses, payables and net worth, your chart better follow this scheme. The available numbers are 1 through 99999998. Since the 1024 screen is 32 characters wide, I used 3 digit numbers.

With these restrictions in mind, we are ready to get started.

First, NEWDISK a box of diskettes. You will need at least seven, and ten is better. (One SYSTEM, and two or three each GENERAL LEDGER, JOURNAL, and ACCOUNTS PAYABLE.)

Next, write labels for each diskette, so when things start to move you know what's what.

Place your miniFLEX disk in #0. This is the disk you got when you bought your computer. Place the newly NEWDISKed disk marked SYSTEM in #1. Enter:

```
COPY,0,1,.CMD,.OV,.LOW,.SYS
LINX,1.DOS
```

Then build your STARTUP file. The instructions came with your SWTPC disk system, and you're interested mostly in TTYSET. The STARTUP file, or the file it calls in, must end with EXEC,0.BOOKSEG. This loads the end-of-period/end-of-year binary program, the BASIC interpreter, and a housekeeping binary program when the system is booted up.

Now remove the SYSTEM disk from #1 and replace it with the BASE disk (which has all the programs on it). Enter:

```
COPY,0.EXEC.CMD
COPY,0.COPY.CMD
```

Take your miniFLEX disk out of #0 and store it away. Take the BASE disk out of #1 and place it in #0. Place your newly made SYSTEM disk back in #1 and enter:

```
EXEC,0.BOOKS.ICL
```

The system takes over from this point, and puts the system programs on the system disk. It then requests you to put a JOU R AL, GENERAL LEDGER (G/L), and ACCOUNTS PAYABLE (A/P) disk on #1. Do what it says, and then enter Y. NOTE: The programs at this point aren't sophisticated. They wait to fetch a character, ANY CHA ACTER, from the keyboard, (to give you time to change disks), and then they take off again. So change disks BEFORE hitting anything!

Since you will need at least two of each data disk, your first answer to the 'another set' question should be 'Y', (no 'return'; just Y). After you have made as many sets as you need (or can afford), enter N. The system will respond with the familiar +++.

At this point, all the disks have their minimum contents. You will now need your own particular data to flesh out the skeletons. After reading the rest of the instructions, get some paper and write down your chart of accounts and their balances. Don't waste your time trying to calculate your net worth. The computer will do that for you.

Power up your computer and put the SYSTEM disk in #0. Enter 'D'. When the system responds READY #, enter: CHAIN 0.INSTALL.

The first file you will build is the parameter file. We need the highest possible account numbers for each of the six categories: income, receivables, assets, expenses, payables, and net worth. (See the sample chart for an example.)

The next file will be the accounts payables. Before putting an A/P disk in the drive, get a label and write today's date on it. Stick it on the disk. The last thing you want is to get the disks mixed up.

The information requested for A/P is:

ACCT #
DESCRIPTION
BALANCE
AYMENT
DATE DUE.

The first three are self-explanatory. The payment entry is used by the cash required program, and should be the amount you expect to pay on the date due. If the payment field is zero, the program assumes the entire balance is due (not so cool in the case of the mortgage). The date due can be either an MMDD format (325 = March 25th) or a DD format (10 = 10th of every month). If the date due is zero, then the program assumes you don't have to pay this one until you want to, and it ignores the account.

The last two entries (payment and date due) can be null entered (return), in which case they default to zero.

When all the accounts payable have been entered, you exit the routine by pressing 'return' without an account number.

Did you get the data on the A/P disk? Good. Now put one on a general ledger disk. The G/L files are identical to the A/P files, except there aren't any payments or dates due.

When you are finished with the general ledger entries, you exit the routine the same as accounts payable. Since the A/P file is already built, there is no reason to enter the data twice. The program will build the A/P section of the G/L, compute the net worth, and call in the START program. Enter today's date and you're finished.

PLEASE, PLEASE, PLEASE, keep track of which disks are current. The small Avery labels are inexpensive, and worth their weight in gold. I have seen a couple sites update from an old disk to their current disk, creating a collossal mess. Some never recover.

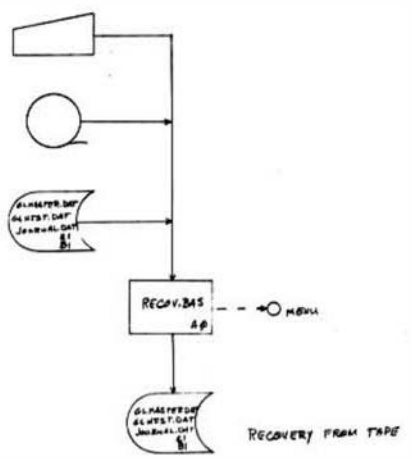
BOOKS.S C

```
                NAM   BOOKS
                OPT   O,S,NOP,NOG
        START   EQU   $2442     SWTPC BASIC 3.0 END
        PDATA   EQU   $E07E     SWTBUG
        INEEE   EQU   $E1AC     SWTBUG
        FLXFCB  EQU   $7809     FLEX 1.0 FCB POINTER
        PBEG    EQU   $A044     SWTBUG
        PEND    EQU   $A004     SWTBUG
        PCHON   EQU   $E34D     SWTBUG
        PUNCH   EQU   $E37E     SWTBUG
        PCHOFF  EQU   $E353     SWTBUG
        DELAY   EQU   $E2C2     SWTBUG
        FMS     EQU   $7806     FLEX 1.0 FMS ENTRY
        BUF     EQU   $7884     FLEX 1.0 DISC BUFFR
        USER    EQU   $5D       BASIC 3.0 USER(X)
        BASIC   EQU   $100      BASIC 3.0 COLD ENTRY
        BASPGM  EQU   $14E      BASIC 3.0 START OF PGMS
        DOSENT  EQU   $32A      BASIC 3.0 JUMP TO DOS
        FLXWRM  EQU   $7103     FLEX 1.0 WARM START
        FLXCLD  EQU   $7100     FLEX 1.0 COLD START
        R   N   EQU   $E334     SWTBUG
        INCH    EQU   $E078     SWTBUG
        CKSM    EQU   $A00F     SWTBUG
        BYTE    EQU   $A055     SWTBUG
        BYTECT  EQU   $A047     SWTBUG
        BADDR   EQU   $E047     SWTBUG
        OUTCH   EQU   $E075     SWTBUG
        RDOFF   EQU   $E347     SWTBUG
                ORG   START
2442            JMP   BOOT      CNTL STNG ENTRY
2445            JMP   ZOJNL     JOURNAL PUNCH
2448            JMP   ZOGL      GL PUNCH
244B            JMP   ZIJNL     JOURNAL READ
244E            JMP   ZIHST     HIST READ
2451            JMP   ZIGL      GL READ
        ZOJNL   LDX   #MSG01    INITIAL M SSAGES
                JSR   PDATA
                LDX   #MSG03
                BRA   OMES1
        ZOGL    LDX   #MSG01
                JSR   PDATA
                LDX   #MSG02
        OMES1   JSR   PDATA
                LDX   #MSG04
                JSR   PDATA
                LDX   #MSG07
                JSR   PDATA
        OMES2   LDX   #MSG08
                JSR   PDATA
                JSR   INEEE
                CMPA  #$D
                BNE   OMES2
                LDAA  FLXFCB+1  GET ADDRESS OF FILE
                LDAB  FLXFCB
                SUBA  #$1C
                SBCB  #0
                STAA  FCB+1
                STAB  FCB
                LDX   #BUF
                STX   CURPOS
                STX   PBEG
                LDAB  #11       GET LABEL READY FOR  UNCH
                LDX   FCB
        OLABEL  LDAA  4,X
                INX
                STX   TEMP
                LDX   CURPOS
                STAA  0,X
                INX
                STX   CU POS
                LDX   TEMP
```

```
710                  DECB
720                  BNE    OLABEL
730                  LDX    CURPOS
740                  DEX
750                  STX    PEND
760                  JSR    PCHON       PUNCH LABEL
770                  JSR    DELAY
780                  JSR    DELAY
790                  JSR    DELAY
800                  JSR    PUNCH
810                  LDX    #69         TO EXIT READ ON RECOV
820                  JSR    PDATA
830                  JSR    PCHOFF
840                  LDX    FCB         PREPARE FILE TO READ
850                  LDAA   #5
860                  STAA   0,X         REWIND
870                  JSR    FMS
880         PREC1    LDX    #BUF
890                  STX    CURPOS
900         PCHR1    LDX    FCB         READ FILE
910                  JSR    FMS
920                  TST    1,X
930                  BNE    OERR        READ ERROR OR EOF
940                  LDX    CURPOS
950                  STAA   0,X
960                  INX
970                  STX    CURPOS
890                  CPX    #BUF+125    END OF BUF?
990                  BNE    PCHR1
1000        PCHR2    LDX    #BUF        PREPARE TO PUNCH DATA
1010                 STX    PBEG
1020                 LDX    CURPOS
1030                 DEX
1040                 STX    PEND
1050                 JSR    PCHON       PUNCH
1060                 JSR    DELAY
1070                 JSR    DELAY
1080                 JSR    DELAY
1090                 JSR    PUNCH
1100                 LDX    #69         TO EXIT READ ON RECOV
1110                 JSR    PDATA
1120                 JSR    PCHOFF
1130                 LDX    FCB
1140                 TST    1,X         LAST BLOCK?
1150                 BEQ    PREC1       GET NEXT RECORD
1160                 BRA    ONOERR      MUST BE EOF
1170        OERR     LDAB   1,X         ERR OR EOF
1180                 CMPB   #8
1190                 BEQ    PCHR2       PUNCH PARTIAL SECTOR
1200        ONOERR   LDX    FCB
1210                 LDAA   1,X
1220                 LDX    USER

1240                 CLRB
1250                 TSTA
1260                 BEQ    OUSER4      NO ERRORS
1270                 CMPA   #8
1280                 BNE    OUSER2      NOT EOF
1290                 CLRA               EOF WILL BE "0" to BASIC
1300        OUSER2   ADDA   #0          CHANGE HEX TO BCD
1310                 DAA
1320                 BITA   #$F0        FORMAT & STORE IN USER(X)
1330                 BEQ    OUSER4
1340                 TAB
1350                 LSRA
1360                 LSRA
1370                 LSRA
1380                 LSRA
1390                 ASLB
1400                 ASLB
1410                 ASLB
1420                 ASLB

1440        OUSER4   CLR    6,X         EXPONENT
1450                 STAA   0,X         SIGN + 1st DIGIT
1460                 BEQ    OUSER5
1470                 INC    6,X         EXPONENT
1480        OUSER5   STAB   1,X         2ND DIGIT
1490                 BEQ    OUSER6
1500                 INC    6,X         EXPONENT
1510        OUSER6   INX
1520                 LDAB   #4          CLEAR REST OF USER(X)
1530        OUSER1   CLR    1,X
1540                 INX
1550                 DECB
1560                 BNE    OUSER1
1570                 LDX    #BUF
1580                 LDAA   #4          EOT FOR LAST TAPE RECORD
1590                 STAA   0,X
1600                 STAA   1,X
1610                 STX    PBEG        PUNCH EOT
1620                 INX
1630                 STX    PEND
1640                 JSR    PCHON
1650                 JSR    DELAY
1660                 JSR    DELAY
1670                 JSR    DELAY
1680                 JSR    PUNCH
1690                 LDX    #69
1700                 JSR    PDATA
1710                 JSR    PCHOFF
1720                 RTS
1730        ZUNL     LDX    #MSG01      INITIAL MESSAGES
1740                 JSR    PDATA
1750                 LDX    #MSG03
1760                 BRA    IMES1
1770        ZIHST    LDX    #MSG01
1780                 JSR    PDATA
1790                 LDX    #MSG02
1800        IMES1    JSR    PDATA
1810                 LDX    #MSG04
1820                 JSR    PDATA
1830                 LDX    #MSG06
1840                 JSR    PDATA
1850        IMES2    LDX    #MSG08
1860                 JSR    PDATA
1870                 JSR    INEEE
1880                 CMPA   #$D
1890                 BNE    IMES2
1900        ZIGL     LDAA   FLXFCB+1    ENTRY FOR NO MESSAG
1910                 LDAB   FLXFCB      GET ADDRESS OF FIL
1920                 SUBA   #$1C
1930                 SBCB   #0
1940                 STAA   FCB+1
1950                 STAB   FCB
1960                 JSR    LOAD        1ST RECORD OF TAPE
1970                 LDX    FCB
1980                 STX    TEMP
1990                 LDX    #BUF
2000                 STX    CURPOS
2010                 LDAB   #11         VERIFY LABEL
2020        LAB1     LDX    TEMP
2030                 LDAA   4,X
2040                 INX
2050                 STX    TEMP
2060                 LDX    CURPOS
2070                 CMPA   0,X
2080                 BNE    BADLAB      WRONG TAPE
2090                 INX
2100                 STX    CURPOS
2110                 DECB
2120                 BNE    LAB1
2130                 LDX    FCB         LAB L OK
2140                 LDAA   #4
2150                 STAA   $22,X       RESET DATA INDEX
     2160 - 2220 deleted
2230        IREC     LDX    #BUF        FILL BUFFER FOR END DETECT
2240                 LDAA   #$FF
2250                 LDAB   #124
2260        IFILL    STAA   0,X
2270                 INX
2280                 DECB
2290                 BNE    IFILL
2300                 JSR    LOAD        READ TA E RECORD
2310                 LDX    #BUF
2320        ICHR     LDAA   0,X
2330                 CMPA   #$FF         ND OF BLOCK
2340                 BEQ    IREC
2350                 CMPA   #4          END OF FILE
2360                 BEQ    IDONE
2370                 INX
2380                 STX    CURPOS
2390                 LDX    FCB
2400                 JSR    FMS         WRITE CHARACTER
2410                 LDAA   1,X
2420                 BNE    IERR        WRITE ERROR: ABORT
2430                 LDX    CURPOS
2440                 CPX    #BUF+125
2450                 BNE    ICHR        GET NEXT CH R
2460                 BRA    IREC        GET NEXT RECORD
2470        BADLAB   LDAA   #8          8 = BAD LABEL
2480                 BRA    ICLO1
2490        IERR     LDX    FCB
2500                 LDAA   1,X
2510                 BRA    ICLO1
2520        IDONE    CLRA
2530        ICLO1    LDX    USER        SET UP USER(X)

2550                 CLRB

2580                 ADDA   #0          HEX TO BCD
2590                 DAA
2600                 BITA   #$F0
2610                 BEQ    ICLO3
2620                 TAB
2630                 LSRA
2640                 LSRA
2650                 LSRA
2660                 LSRA
2670                 ASLB
2680                 ASLB
2690                 ASLB
2700                 ASLB

2720        ICLO3    CLR    6,X         EXPONENT
2730                 STAA   0,X         SIGN + 1ST DIGIT
2740                 BEQ    ICLO4
2750                 INC    6,X         EXPONENT
2760        ICLO4    STAB   1,X         2ND & 3RD DIGITS
2770                 BEQ    ICLO5
2780                 INC    6,X         EXPONENT
2790        ICLO5    INX
2800                 LDAB   #4          CLEAR REST OF USER(X)
2810        TCLO1    CLR    1,X
2820                 INX
2830                 DECB
2840                 BNE    TCLO1
2850                 RTS
```

```
        MSG01  FDB   $1016,0,0
               FCC   /THE /
               FCB   4
        MSG02  FCC   /GENERAL LEDGER/
               FCB   4
        MSG03  FCC   /JOURNAL/
               FCB   4
        MSG04  FDB   $D0A
               FCC   /IS READY FOR TAPE./
               FCB   $D,$A,$A
               FCC   /SET BAUD AT 300 FOR K. C. STD./
               FDB   $D0A
               FCC   /TAPE./
               FCB   $D,$A,$A
               FCC   /PUT TAPE IN RECORDER AND/
               FDB   $D0A
               FCC   /PREPARE TO /
               FCB   4
        MSG06  FCC   /READ./
               FCB   4
        MSG07  FCC   /RECORD./
               FCB   4
        MSG08  FCB   $D,$A,$A
               FCC   /RETURN WHEN READY.      ? /
               FCB   4
        SS     FCC   /SS/
               FCB   4
        TEMP   RMB   2
        CURPOS RMB   2
        FCB    RMB   2
        LOAD   JSR   RDON      SWTBUG READ MODIFIED
        LOAD3  JSR   INCH      AS SUBROUTINE
               CMPA  #'S
               BNE   LOAD3
               JSR   INCH
               CMPA  #'9
               BEQ   LOAD21
               CMPA  #'1
               BNE   LOAD3
               CLR   CKSM
               JSR   BYTE
               SUBA  #2
               STAA  BYTECT
               JSR   BADDR
        LOAD11 JSR   BYTE
               DEC   BYTECT
               BEQ   LOAD15
               STAA  0,X
               CMPA  0,X
               BNE   LOAD19
               INX
               BRA   LOAD11
        LOAD15 INC   CKSM
               BEQ   LOAD3
        LOAD19 LDAA  #'?
               JSR   OUTCH
        LOAD21 JSR   RDOFF
               RTS
        DOS    LDX   #FLXWRM   FIRST FLEX ENTR  MUST
               STX   DOSENT    BE COLD
               JMP   FLXCLD
        BOOT   LDX   #DOS      INITIALIZE BASIC
               STX   DOSENT
               LDX   #BOOT
               STX   BASPGM
               JMP   BASIC
2780    DONE   FDB   *         FIND END OF PGM FROM
               END             SYMBOL TABLE
```

NOTE: Object file = BOOKS.BIN,2442,277F,2442



RECOVERY FROM TAPE

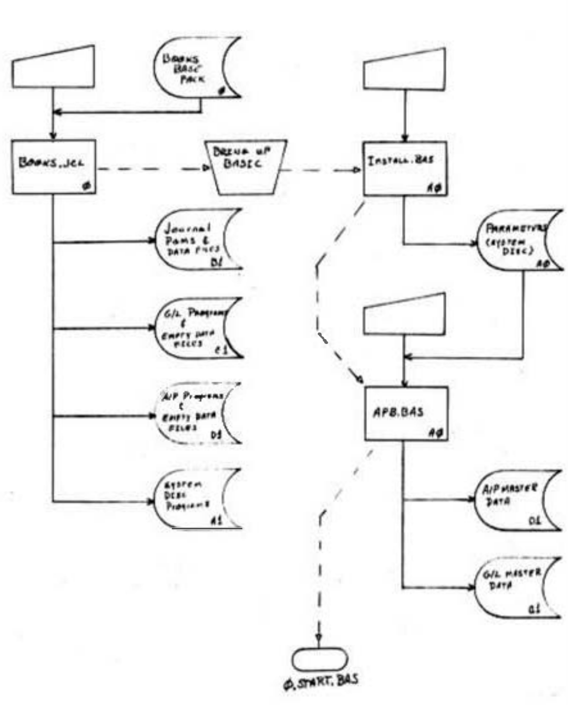
SAMPLE CHART OF ACCOUNTS

111	Dad's income
121	Mom's income
131	XYZ capital gains
132	ABQ capital gains
141	golden savings interest
142	rog savings interest
151	XYZ dividends
152	ABQ dividends
200 - 299	accts receivable (bal)
311	XYZ stock
312	ABQ stock
321	golden savings
322	rog savings
323	wife's savings (double ha)
324	checking
331	house
332	cars
333	furniture &c.
334	appliances
335	other (computer)
412	interest on house note
413	house insurance
414	house taxes
421	gas & elect
422	water
423	phone
424	elect maint
425	plumbing maint
426	heating maint
427	general repairs
429	other
431	furnishings under $100
432	furnishings over $100
491	books, magazines...
511	Dad's allowance
512	Mom's allowance
520	food
530	babysitters
540	tuition
550	medical
560	personal insurance
571	Dad's clothing
572	Mom's clothing
573	kid #1 clothing
574	kid #2 clothing
575	kid #3 clothing
581	Christmas gifts
582	other gifts
583	united appeal &c.
584	Cr card interest $$$
585	church donations
586	Merrill Lynch donations
589	misc. misc.
611	Dad's gas and oil
612	maintenance
613	insurance
614	company reimbursed
621	wife's gas & oil
622	wife's car maint
623	wife car ins
711	dad fed withhold
712	dad fica
713	dad state donations
721	wife fed withhold
722	wife fica
723	wife state tax

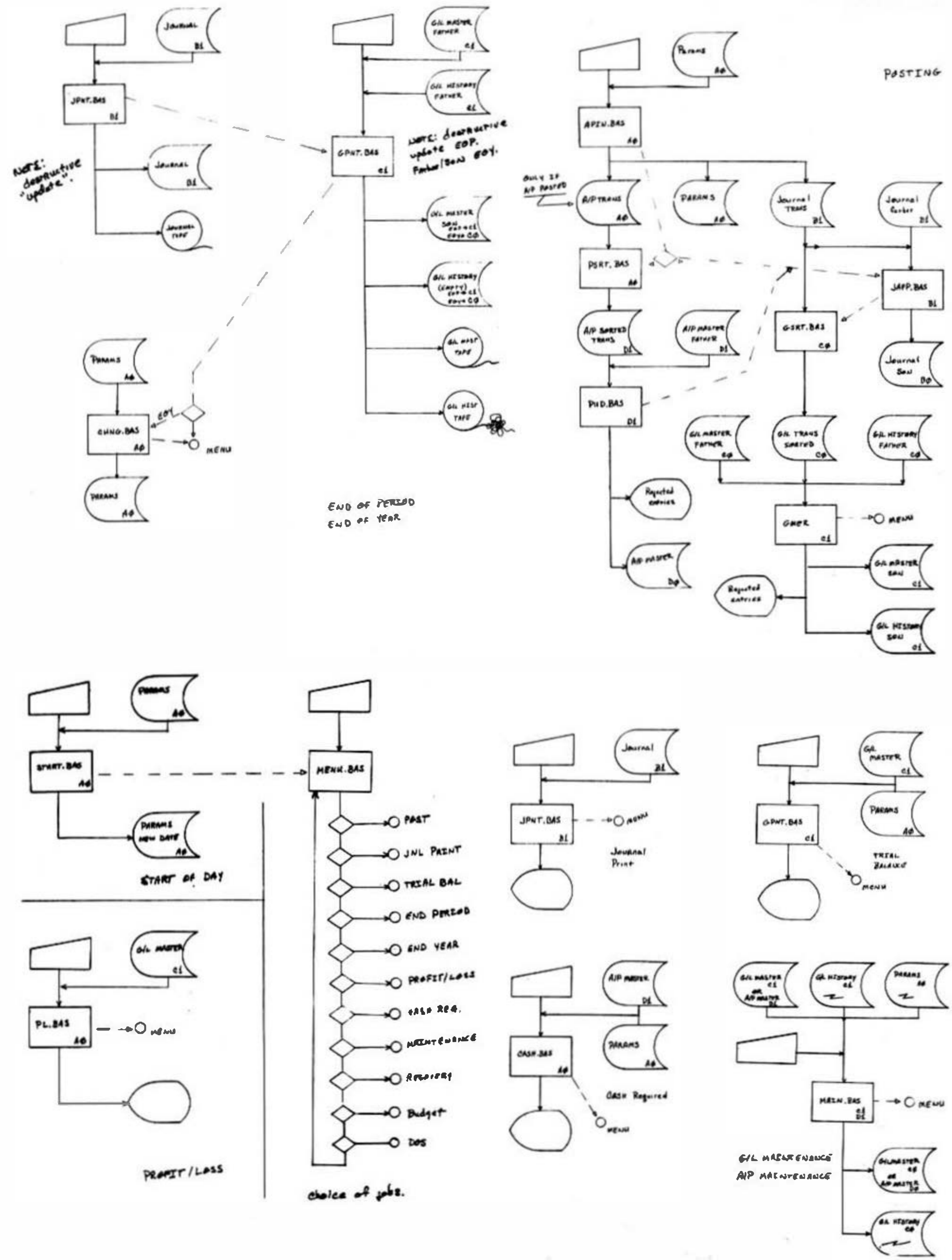
SAMPLE CHART OF ACCOUNTS (cont)

811	DEF insurance		
812	credit card #1	INCOME	199
813	mortgage	RECEIVABLES	299
814	dept store #1	ASSETS	399
815	dept store #2	EXPENSES	799
816	credit card #2	PAYABLES	899
817	credit card #3	NET WORTH	900
...			
900	Net Worth		

(Columns 3–4: Sample entries for parameter build, using the above chart:)



CONVERSION


JPNT.BAS
GPNT.BAS
APIN.BAS
PSRT.BAS
PUD.BAS
GSRT.BAS
JAPP.BAS
CHNG.BAS
POSTING
END OF PERIOD
END OF YEAR
START.BAS
MENU.BAS
START OF DAY
PL.BAS
PROFIT/LOSS
PAST
JNL PRINT
TRIAL BAL
END PERIOD
END YEAR
PROFIT/LOSS
MAINTENANCE
Budget
DOS
choice of jobs.
Journal Print
CASH.BAS
CASH Required
MAIN.BAS
G/L MAINTENANCE
A/P MAINTENANCE

A SOFTWARE DATA ENCRYPTON STANDARD IMPLEMENTATION FOR THE 6800

S. J. Lacour and T. F. Elbert
The University of West Florida
Pensacola, FL 32504

The security of digital communications is becoming of great importance as the use of distributed systems becomes commonplace. Several systems for encrypting sensitive data have recently appeared in the literature, and some controversy has arisen over the relative security of these systems. The purpose of this article is to describe a 6800 software implementation of one of these systems, the National Bureau of Standards Data Encryption Standard (DES). This particular standard was developed by NBS in response to the requirement for a single certifiable standard to be used for all federal government unclassified data stored and transmitted by computer.

The contract to develop the standard was issued to IBM in 1974. During the development phase of the program, the National Security Agency was consulted regarding certain aspects of the standard, one of which was the key length. This fact led to speculation that perhaps the NSA had "tampered" with the encryption algorithm, creating a weakness which only they could exploit. The NBS Data Encryption standard was adopted on November 23, 1976, with an effective date of July 15, 1977. After this date, all federal agencies were required to comply with the standard. On April 13, 1978, the United States Senate Select Committee on Intelligence issued a report which, among other things, concluded that NSA did not tamper with the design of the DES algorithm in any way. And so, the DES exists today as the single method by which encryption of nonclassified data within all federal agencies is accomplished.

The Data Encryption Standard is thoroughly described in Federal Information Processing Standards (FIPS) Publication 46, U.S. Department of Commerce, National Bureau of Standards, issued on January 15, 1979. It is described as "an algorithm to be implemented in electronic hardware devices" and not by software. The 6800 user who can utilize the Motorola Exorcisor bus configuration can purchase a data security module which will encrypt a 64-bit block of data in less than 200 microseconds, and which has been certified by the NBS. The cost is around $500. Those computer users with a requirement for data encryption not involving any federal agency, or those merely wishing to experiment with data encryption, can use a software implementation of the algorithm. Such software implementations offer the same immunity to cryptanalysis as the hardware versions, in that the best machines available for the next few years would take some 200 years to break the code.

Since the DES algorithm is fully explained in FIPS Publication 46, only the rudiments will be discussed here. The algorithm utilizes a 64 bit input block, a 64 bit outblock block, and a 64 bit key of which 56 bits are actually used as the key, with the remaining eight bits being reserved for parity checks on the key itself. The 64 bit input block is first passed through an initial permutation (IP) which shuffles the input bits in accordance with a specified permutation table. The resulting 64 bit permuted input is then split into two 32 bit blocks, L and R, such that the permuted input block is LR. The L and R blocks are then passed through 16 iterations of a calculation described below in terms of a cipher function f. Successive functions L_m and R_m are determined by the recursive equations

$$L_m = R_{m-1}$$
$$R_m = L_{m-1} \oplus f\,(R_{m-1}, K_m)$$

Where initial values L_0 and R_0 are those resulting from initial division of the permuted input block, and where $\oplus$ represents the bit-by-bit exlusive OR operation. The subkey K_m is a block of 48 bits chosen from the 64 bit key in accordance with the expression

$$K_m = KS\ (M, KEY)$$

where KEY is the 64 bit input key, KS is a function called the key schedule, and K_m is determined by the bits in 48 distinct bit positions within KEY, as specificed by the key schedule. The KS function consists of putting the 64 bit key through a specified permutation and bit selection process (PC-1), resulting in two 28 bit blocks termed C_0 and D_0. These blocks are then left shifted in accordance with a specified schedule to generate C_m and D_m for each of the sixteen iterations. The block $C_m D_m$ is then passed through a second permutation and bit selection process (PC-2) to produce K_m, a 48 bit block used as the subkey for iteration M.

Finally, the cipher function $f(R_{m-1}, K_m)$ is determined by first forming a 48 bit function $E(R_{m-1})$ from the 32 bit R_{m-1} block by means of a specified bit selection table, exclusive OR-ing this block with the 48 bit subkey KM, and then passing each six bit block of the result through a specified selection function, $S_1, S_2, \ldots, S_8$. These so called "S-boxes" generate eight four bit blocks, one from each of the six bit blocks, which combine to form a 32 bit result. This result then undergoes a final permutation P to produce the 32 bit cipher function $f(R_{m-1}, K_m)$. The 32 bit R_m is then determined from f and L_{m-1} as described above.

When L_{16} and R_{16} are finally determined, the 64 bit block $L_{16}R_{16}$ is passed through the inverse of the initial permutation (IP^{-1}) to yield the 64 bit output ciphertext. To decrypt a ciphertext encoded by the DES algorithm, it is necessary only to process the encrypted block through the same algorithm, only now the subkeys K_n are generated in reverse order.

The algorithm itself is fairly complex but the procedures, including the permutations, bit selections, shift schedules, and the S-boxes themselves, are public knowledge and are fully described in FIPS Publication 46. The only thing which needs to be kept secret is the key. The reasons behind the particular selection of these various functions by NBS is not obvious to those unfamiliar with cryptographic techniques, but it must be assumed that the selections and procedures were chosen to enhance the security of the algorithm against cryptanalysis. In fact, it is the design of the S-boxes themselves, which has never been explained by IBM, NBS, or NSA, which has led to the speculation of tampering.

The particular software version of the DES described below was written for a SWTP 6800 microcomputer using the FLEX operating system, and requires approximately 1100 bytes of memory. The permutation routine PERM is used most often and with various inputs, outputs, and permutation tables. A parameter table is used to indicate to the permutation routine the number of bits in the output byte, the number of output bytes, the location of input and output blocks, and the desired permutation table. All permutation tables are stored in the format mmmmbbbb as described in the program listing. The mask number mmmm gives the location of the source bit within the source byte. The bbbb gives the location within the input of the byte which contains the desired bit. The desired bit is masked out and shifted into a holding byte which, when full, is stored in the output block.

Routine SHIFT generates the shifting operation necessary for generation of the subkeys. Since each subkey corresponds to a specific iteration of L and R, it is used only once in the encryption of a given 64 bit input block. This makes it possible for the subkeys to be generated as they are needed, rather than having them stored in memory. This requires a shift routine which will handle both left and right shifting of 28 bit blocks of data.

The routine which requires the most memory and table searching is PERMS, which performs the S-box mapping. Since each element of an S-box can fit into a half byte, a compacted table is used so that two elements are contained in a single byte. To access an S-box entry, the row and column numbers are specified. The column number is then divided by two and added to the row number, which in turn is adjusted to the left half byte of the table pointer. This provides the table offset, which is added to the table address to get the byte containing the desired half-byte. The particular half-byte of interest is then determined, together with the specification of the half of the output byte into which it is to be stored.

The routine which actually performs the iterative procedures of the DES algorithm is ITER. It calls the various subroutines, and directs the logic flow for both encryption and decryption. It follows the iterative procedures described above for generating L_m, R_m, and K_m for each of the 16 iterative steps.

The main routine converts the DES program into a FLEX utility responding to the entry DES from a FLEX prompt. It responds with a user prompt for the DES parameters, and calls ITER to initiate the encryption or decryption. The DES parameters are:

(1) MODE (00 = encrypt, 01 = decrypt)
(2) KEY (16 hexadecimal digits)
(3) INPUT (16 hexadecimal digits)

After the input data is entered, the system responds with the output of 16 hexadecimal

digits. A sample encryption followed by a decryption is shown following the program listing.

It will be noted that the main routine utilizes certain FLEX and MIKBUT (DISKBUG in this case) routines. These are not necessary unless something like the main routine is used to interface the program with FLEX. Also, the FLEX routine ADDBX is used by two of the subroutines. This merely adds the B accumulator to the index register, and could be replaced by a user provided subroutine. It is used to provide a variable offset to the index register, permitting easier table access.

REFERENCES

FIPS Publication 46, "Data Encryption Standard," U.S. Department of Commerce, Bureau of Standards, January 15, 1977.

NBS Special Publication 550-20 "Validating the Correctness of Hardware." "Implementations of the NBS Data Encryption Standard," U.S. Department of Commerce, Bureau of Standards, November, 1977.

Mueshaw, R. V. "The Standard Data Encryption Algorithm, Part 1," BYTE, March, 1979.

Mueshaw, R. V. "The Standard Data Encryption Algorithm, Part 2," BYTE, April, 1979.

Davis, Ruth. "The Data Encryption Standard in Perspective," IEEE Communications Society Magazine, November, 1978.

"Unclassified Summary: Involvement of NSA in the Development of the Data Encryption Standard," U.S. Select Committee on Intelligence, IEEE Communications Society Magazine, November, 1978.


```
                    NAM     DES
          ****************************************************
          *     STANDARD DATA ENCRYPTION ALGORITHM           *
          ****************************************************
          *          AUTHOR: Sam J. La Cour Jr.              *
          ****************************************************
0000                ORG     $0000
0000 20 01          BRA     ENT
0002 01    VN       FCB     1
0003 7E 03 19 ENT   JMP     MAIN
0006 45    CODMSG   FCC     /ENTER A 00 FOR ENCRYPT, 01 FOR DECRYPT /
002D 04             FCB     $04
002E 49    KEYMSG   FCC     /INPUT KEY (8 BYTES)/
0041 04             FCB     $04
0042 45    INPMSG   FCC     /ENTER DATA TO BE ENCODED (8 BYTES)/
0064 04             FCB     $04
0065       STLOC    RMB     2
0067       NRESBT   RMB     1
0068       NPERM    RMB     1
0069       AINPUT   RMB     2
006B       ARESWD   RMB     2
006D       TABADR   RMB     2
006F       NBIT     RMB     1
0070       ED       RMB     1
0071       TMASK    RMB     1
0072       L        RMB     4
0076       R        RMB     4
007A       K        RMB     8
0082       K1       RMB     4
0086       INPUT    RMB     8
008E       C        RMB     4
0092       D        RMB     4
0096       SD16     RMB     1
0097 3F    [illegible] FCB  $3F,$7E
0099 00 7A KADR     FDB     K
009B 00 82 K1ADR    FDB     K1
009D 01 D9 SADR     FDB     S
009F       KEY      RMB     8
00A7       APARM    RMB     2
00A9 08    PARMS    FCB     $08,$08
00AB 00 86          FDB     INPUT,L,INITAB
00B1 08             FCB     $08,$04
00B3 00 9F          FDB     KEY,C,PCH1A
00B9 08             FCB     $08,$04
00BB 00 9F          FDB     KEY,D,PCH1B
00C1 06    KPARM    FCB     $06,$08
00C3 00 8E          FDB     C,K,PCH2
00C9 06             FCB     $06,$08
00CB 00 76          FDB     R,INPUT,E
00D1 08             FCB     $08,$04
00D3 00 82          FDB     K1,INPUT,P
00D9 08             FCB     $08,$08
00DB 00 72          FDB     L,INPUT,INVTAB
00E1 27    INITAB   FCB     $27,$26,$25,$24,$23,$22,$21,$20
00E9 47             FCB     $47,$46,$45,$44,$43,$42,$41,$40
00F1 67             FCB     $67,$66,$65,$64,$63,$62,$61,$60
00F9 87             FCB     $87,$86,$85,$84,$83,$82,$81,$80
0101 17             FCB     $17,$16,$15,$14,$13,$12,$11,$10
0109 37             FCB     $37,$36,$35,$34,$33,$32,$31,$30
0111 57             FCB     $57,$56,$55,$54,$53,$52,$51,$50
0119 77             FCB     $77,$76,$75,$74,$73,$72,$71,$70
0121 17    PCH1A    FCB     $17,$16,$15,$14,$13,$12,$11
0128 10             FCB     $10,$27,$26,$25,$24,$23,$22
012F 21             FCB     $21,$20,$37,$36,$35,$34,$33
0136 32             FCB     $32,$31,$30,$47,$46,$45,$44
013D 77    PCH1B    FCB     $77,$76,$75,$74,$73,$72,$71
0144 70             FCB     $70,$67,$66,$65,$64,$63,$62
014B 61             FCB     $61,$60,$57,$56,$55,$54,$53
0152 52             FCB     $52,$51,$50,$43,$42,$41,$40
0159 61    PCH2     FCB     $61,$12,$31,$82,$10,$50
015F 30             FCB     $30,$43,$71,$60,$52,$21
0165 72             FCB     $72,$32,$41,$40,$23,$80
016B 81             FCB     $81,$70,$33,$42,$51,$20
0171 55             FCB     $55,$86,$34,$15,$36,$37
0177 24             FCB     $24,$45,$76,$16,$54,$46
017D 85             FCB     $85,$56,$35,$47,$64,$17
0183 26             FCB     $26,$65,$66,$84,$14,$44
0189 83    E        FCB     $83,$10,$20,$30,$40,$50
018F 40             FCB     $40,$50,$60,$70,$80,$11
0195 80             FCB     $80,$11,$21,$31,$41,$51
019B 41             FCB     $41,$51,$61,$71,$81,$12
01A1 81             FCB     $81,$12,$22,$32,$42,$52
01A7 42             FCB     $42,$52,$62,$72,$82,$13
01AD 82             FCB     $82,$13,$23,$33,$43,$53
01B3 43             FCB     $43,$53,$63,$73,$83,$10
01B9 81    P        FCB     $81,$70,$42,$52
01BD 53             FCB     $53,$41,$43,$12
01C1 10             FCB     $10,$71,$72,$23
01C5 50             FCB     $50,$22,$73,$21
01C9 20             FCB     $20,$80,$82,$61
01CD 83             FCB     $83,$33,$30,$11
01D1 32             FCB     $32,$51,$63,$60
01D5 62             FCB     $62,$31,$40,$13
01D9 E4    S        FCB     $E4,$D1,$2F,$B8,$3A,$6C,$59,$07
01E1 0F             FCB     $0F,$74,$E2,$D1,$A6,$CB,$95,$38
01E9 41             FCB     $41,$EB,$D6,$2B,$FC,$97,$3A,$50
01F1 FC             FCB     $FC,$82,$49,$17,$5B,$3E,$A0,$6D
01F9 F1             FCB     $F1,$8E,$6D,$34,$97,$2D,$C0,$5A
0201 3D             FCB     $3D,$47,$F2,$8E,$C0,$1A,$69,$B5
0209 0E             FCB     $0E,$7B,$A4,$D1,$58,$C6,$93,$2F
0211 D8             FCB     $D8,$A1,$3F,$42,$B6,$7C,$05,$E9
0219 A0             FCB     $A0,$9E,$63,$F5,$1D,$C7,$B4,$28
0221 D7             FCB     $D7,$09,$34,$6A,$28,$5E,$C3,$F1
0229 D6             FCB     $D6,$49,$8F,$30,$B1,$2C,$5A,$E7
0231 1A             FCB     $1A,$D0,$69,$87,$4F,$E3,$B5,$2C
0239 7D             FCB     $7D,$E3,$06,$9A,$12,$85,$BC,$4F
0241 D8             FCB     $D8,$B5,$6F,$03,$47,$2C,$1A,$E9
0249 A6             FCB     $A6,$90,$CD,$7D,$F1,$3E,$52,$84
0251 3F             FCB     $3F,$06,$A1,$D8,$94,$5B,$C7,$2E
0259 2C             FCB     $2C,$41,$7A,$B6,$85,$3F,$D0,$E9
0261 EB             FCB     $EB,$2C,$47,$D1,$50,$FA,$39,$86
0269 42             FCB     $42,$1B,$AD,$78,$F9,$C5,$63,$0E
0271 B8             FCB     $B8,$C7,$1E,$2D,$6F,$09,$A4,$53
0279 C1             FCB     $C1,$AF,$92,$68,$0D,$34,$E7,$5B
0281 AF             FCB     $AF,$42,$7C,$95,$61,$DE,$0B,$38
0289 9E             FCB     $9E,$F5,$28,$C3,$70,$4A,$1D,$B6
0291 43             FCB     $43,$2C,$95,$FA,$BE,$17,$60,$8D
0299 4D             FCB     $4D,$2E,$F0,$8D,$3C,$97,$5A,$61
02A1 D0             FCB     $D0,$B7,$49,$1A,$E3,$5C,$2F,$86
02A9 14             FCB     $14,$BD,$C3,$7E,$AF,$68,$05,$92
02B1 6B             FCB     $6B,$D8,$14,$A7,$95,$0F,$E2,$3C
02B9 D2             FCB     $D2,$B4,$6F,$D1,$A9,$3E,$50,$C7
02C1 1F             FCB     $1F,$D8,$A3,$74,$C5,$6B,$0E,$92
02C9 7B             FCB     $7B,$41,$9C,$E2,$06,$AD,$F3,$58
02D1 21             FCB     $21,$E7,$4A,$8D,$FC,$90,$35,$6B
02D9 84    INVTAB   FCB     $84,$80,$85,$81,$86,$82,$87,$83
02E1 74             FCB     $74,$70,$75,$71,$76,$72,$77,$73
02E9 64             FCB     $64,$60,$65,$61,$66,$62,$67,$63
02F1 54             FCB     $54,$50,$55,$51,$56,$52,$57,$53
02F9 44             FCB     $44,$40,$45,$41,$46,$42,$47,$43
0301 34             FCB     $34,$30,$35,$31,$36,$32,$37,$33
0309 24             FCB     $24,$20,$25,$21,$26,$22,$27,$23
0311 14             FCB     $14,$10,$15,$11,$16,$12,$17,$13
E1AC       INEEE    EQU     $E1AC
E1D1       OUTEEE   EQU     $E1D1
E0BF       OUT2H    EQU     $E0BF
E0C8       OUT4HS   EQU     $E0C8
E047       BADDR    EQU     $E047
E055       BYTE     EQU     $E055
E07E       PDATA1   EQU     $E07E
E0CA       OUT2HS   EQU     $E0CA
E0CC       OUTS     EQU     $E0CC
AD03       WARMS    EQU     $AD03
AD1E       PSTRNG   EQU     $AD1E
AD2D       GETFIL   EQU     $AD2D
AD33       SETEXT   EQU     $AD33
AD36       ADDBX    EQU     $AD36
AD3F       RPTERR   EQU     $AD3F
```

```
AD15                GETCHR EQU   $AD15
AD24                PCRLF  EQU   $AD24
AD18                PUTCHR EQU   $AD18
                    ***********************************************
                    * MAIN ROUTINE:                               *
                    *                                             *
                    *  FUNCTION: Prompts user for DES parameters and *
                    *            invokes ITER to do the encryption *
                    *            or decryption. The DES parameters *
                    *            consist of:                      *
                    *              (1) MODE (00=Encrypt,01=Decrypt) *
                    *              (2) KEY  (8 hex bytes).        *
                    *              (3) INPUT DATA (8 hex bytes).  *
                    *                                             *
                    *  EXTERNAL ROUTINES:                         *
                    *                                             *
                    *     FLEX: PCRLF, WARMS                      *
                    *     DISKBUG: BYTE, PDATA1, OUT2HS           *
                    *                                             *
                    ***********************************************
0319 BD AD 24       MAIN   JSR   PCRLF
031C CE 00 06              LDX   #MODMSG
031F BD E0 7E              JSR   PDATA1
0322 BD AD 24              JSR   PCRLF
0325 BD E0 55              JSR   BYTE
0328 97 96                 STA A SDIR
032A BD AD 24              JSR   PCRLF
032D CE 00 2E              LDX   #KEYMSG
0330 BD E0 7E              JSR   PDATA1
0333 BD AD 24              JSR   PCRLF
0336 C6 08                 LDA B #08
0338 CE 00 9F              LDX   #KEY
033B 37             INKEY  PSH B
033C BD E0 55              JSR   BYTE
033F A7 00                 STA A 0,X
0341 08                    INX
0342 33                    PUL B
0343 5A                    DEC B
0344 2E F5                 BGT   INKEY
0346 BD AD 24              JSR   PCRLF
0349 CE 00 42              LDX   #INPMSG
034C BD E0 7E              JSR   PDATA1
034F BD AD 24              JSR   PCRLF
0352 C6 08                 LDA B #08
0354 CE 00 86              LDX   #INPUT
0357 37             INPLO  PSH B
0358 BD E0 55              JSR   BYTE
035B A7 00                 STA A 0,X
035D 08                    INX
035E 33                    PUL B
035F 5A                    DEC B
0360 2E F5                 BGT   INPLO
0362 BD AD 24              JSR   PCRLF
0365 BD 03 B3              JSR   ITER
0368 C6 08                 LDA B #08
036A CE 00 86              LDX   #INPUT
036D BD E0 CA       PX     JSR   OUT2HS
0370 5A                    DEC B
0371 2E FA                 BGT   PX
0373 7E AD 03              JMP   WARMS
                    ***********************************************
                    * SUBROUTINE SHIFT:                           *
                    *                                             *
                    *  FUNCTION: Rotates 28 bits pointed to by X  *
                    *            either left or right depending on *
                    *            the value in "SDIR" (00 for left, *
                    *            01 for right). The result is stored *
                    *            in the source field.             *
                    *                                             *
                    *  EXTERNAL ROUTINES: NONE                    *
                    *                                             *
                    ***********************************************
0376 36             SHIFT  PSH A             Save accumulators A and B.
0377 37                    PSH B
0378 36                    PSH A             Clear out right side of last
0379 A6 03                 LDA A 3,X         of C or D.
037B 84 F0                 AND A #$F0
037D A7 03                 STA A 3,X
037F 32                    PUL A
0380 81 01                 CMP A #$01        Encrypt or decrypt?
0382 27 15                 BEQ   RSHIF1      Decrypt -- right rotate.
0384 68 03          LSHIF1 ASL   3,X         Encrypt -- left rotate.
0386 A6 03                 LDA A 3,X
0388 69 02                 ROL   2,X
038A 69 01                 ROL   1,X
038C 69 00                 ROL   0,X
038E 89 0F                 ADC A #$0F        Force carry into bit 4 of 1
0390 84 F0                 AND A #$F0        byte and reset bits 3 - 0.
0392 A7 03                 STA A 3,X
0394 5A                    DEC B             More than one shift?
0395 2E ED                 BGT   LSHIFT      Yes -- do it again.
0397 20 17                 BRA   OUTSHF      Exit.
0399 64 00          RSHIF1 LSR   0,X         Decrypt -- right rotate.
039B 66 01                 ROR   1,X
039D 66 02                 ROR   2,X
039F 66 03                 ROR   3,X
03A1 A6 03                 LDA A 3,X
03A3 48                    ASL A             Force previous bit 4 of last byte
03A4 48                    ASL A             to position 1 so it can be placed
03A5 48                    ASL A             in bit 1 of the first byte which
03A6 48                    ASL A             has been cleared by the LSR 0,X.
03A7 84 80                 AND A #$80        Get this bit.
03A9 AA 00                 ORA A 0,X         OR it into byte 1.
03AB A7 00                 STA A 0,X         Resave byte 1.
03AD 5A                    DEC B             More than one shift?
03AE 2E E9                 BGT   RSHIFT      Yes -- do it again.
03B0 33             OUTSHF PUL B             Restore accumulators A and B.
03B1 32                    PUL A
03B2 39                    RTS
                    ***********************************************
                    * SUBROUTINE ITER:                            *
                    *                                             *
                    *  FUNCTION: Actually the main DES routine, ITER *
                    *            calls all other DES routines and *
                    *            directs logic flow for either    *
                    *            encryption or decryption.        *
                    *                                             *
                    * EXTERNAL ROUTINES: NONE                     *
                    *                                             *
                    ***********************************************
03B3 CE 00 A9       ITER   LDX   #PARMB      Initialize parameter pointer to
03B6 DF 65                 STX   STLOC       first group of parameters.
03B8 CE 3F 7E              LDX   #$3F7E      Initialize shift schedule indicator.
03BB DF 97                 STX   SHUN
03BD BD 04 4F              JSR   PERM        Perform initial permutation.
03C0 BD 04 4F              JSR   PERM        Perform permuted choice 1a.
03C3 BD 04 4F              JSR   PERM        Perform permuted choice 1b.
03C6 86 10                 LDA A #16         Initialize iteration counter.
03C8 36             ITER1  PSH A             Save current iteration counter.
03C9 CE 00 C1              LDX   #KPARM      Force parameter pointer to start
03CC DF 65                 STX   STLOC       of parameters used in iterations.
03CE 96 96                 LDA A SDIR        Get encrypt, decrypt mode.
03D0 26 1B                 BNE   DECR        If 1 then decrypt mode.
03D2 78 00 98              ASL   SHUN+1      Rotate next shift schedule bit into
03D5 79 00 97              ROL   SHUN        carry then add the carry to 1,
03D8 C6 01                 LDA B #$01        giving the number of shifts.
03DA C9 00                 ADC B #$00
03DC CE 00 8E              LDX   #C          Prepare to shift C to the left.
03DF BD 03 76              JSR   SHIFT       Shift C left.
03E2 CE 00 92              LDX   #D          Prepare to shift D to the left.
03E5 BD 03 76              JSR   SHIFT       Shift D left.
03E8 BD 04 4F              JSR   PERM        Perform permuted choice 2.
03EB 20 1A                 BRA   DOON        Continue.
03ED 8D 60          DECR   BSR   PERM        Perform permuted choice 2 first.
03EF 96 96                 LDA A SDIR        Get encrypt, decrypt code.
03F1 76 00 97              LSR   SHUN        Rotate next shift schedule bit into
03F4 76 00 98              ROR   SHUN+1      carry ( from right for decryption ).
03F7 C6 01                 LDA B #$01        Now add the carry to 1 giving the
03F9 C9 00                 ADC B #$00        number of shifts.
03FB CE 00 8E              LDX   #C          Prepare to shift C to the right.
03FE BD 03 76              JSR   SHIFT       Shift C to the right.
0401 CE 00 92              LDX   #D          Prepare to shift D to the right.
0404 BD 03 76              JSR   SHIFT       Shift D to the right.
0407 8D 46          DOON   BSR   PERM        Perform E permutation on R.
0409 CE 00 7A              LDX   #K          Prepare to EOR subkey K with
040C 86 08                 LDA A #08         result of E permutation, result in K.
040E 36             NST1   PSH A
040F A6 00                 LDA A 0,X         Get a byte of K.
0411 A8 0C                 EOR A 12,X        EOR it with a byte of "INPUT".
0413 A7 00                 STA A 0,X         Store it back in K.
0415 08                    INX
0416 32                    PUL A
0417 4A                    DEC A
0418 2E F4                 BGT   NST1
041A BD 04 B1              JSR   PERMS       Perform S1 - S8 selection mapping.
041D 8D 30                 BSR   PERM        Perform P permutation giving f(R,K).
041F CE 00 72              LDX   #L          Prepare to swap and EOR L and R.
0422 86 04                 LDA A #04         Four bytes in each.
0424 36             NST2   PSH A
0425 A6 00                 LDA A 0,X         Get a byte of L.
0427 A8 14                 EOR A 20,X        EOR L with f(R,K) ( in "INPUT" ).
0429 E6 04                 LDA B 4,X         Get a byte of R.
042B A7 04                 STA A 4,X         Put new L where R was.
042D E7 00                 STA B 0,X         Put R where L was.
042F 08                    INX
0430 32                    PUL A
0431 4A                    DEC A
0432 2E F0                 BGT   NST2
0434 32                    PUL A             Pull iteration counter off stack.
0435 4A                    DEC A             Decrement it.
0436 2E 90                 BGT   ITER1       Keep going.
0438 CE 00 72              LDX   #L          Prepare to perform final swap.
043B 86 04                 LDA A #4
043D 36             SW1    PSH A
043E A6 00                 LDA A 0,X         Get a byte of L.
0440 E6 04                 LDA B 4,X         Get a byte of R.
0442 E7 00                 STA B 0,X         Put R in L.
0444 A7 04                 STA A 4,X         Put L in R.
0446 08                    INX
0447 32                    PUL A
0448 4A                    DEC A
0449 2E F2                 BGT   SW1
044B BD 04 4F              JSR   PERM        Perform inverse initial permutation
044E 39                    RTS               Finished.
                    ***********************************************
                    * SUBROUTINE PERM:                            *
                    *                                             *
                    *  FUNCTION: Performs bit mapping from input  *
                    *            to output using a mapping table. *
                    *            Input,output and table are specified *
                    *            in a parameter table which is    *
                    *            traversed serially each time     *
                    *            PERM is called. The mapping table *
                    *            entries are in the form aaaabbbb *
                    *            where aaaa is a mask number which *
                    *            gives the location of the source *
                    *            bit within the source byte.      *
                    *            The bits are numbered from 1 TO 8. *
                    *            bbbb gives the location within the *
                    *            input of the byte which contains *
                    *            the desired bit. These bytes are *
                    *            numbered from 0 TO 7.            *
                    *            The parameter list also gives the *
                    *            number of bytes and the byte length *
                    *            of the output.                   *
                    *                                             *
                    *  EXTERNAL ROUTINES:                         *
                    *     FLEX: ADDBX                             *
                    *                                             *
                    ***********************************************
044F 86 08          PERM   LDA A #$08        Number of parameter bytes to be moved
0451 CE 00 A7              LDX   #PRESET     Address of parameter storage area.
0454 DF A7                 STX   APARM
0456 DE 65          P1     LDX   STLOC       Current parameter pointer.
0458 E6 00                 LDA B 0,X         Get a parameter byte.
045A 08                    INX               Move parameter pointer along.
045B DF 65                 STX   STLOC       Save it.
045D DE A7                 LDX   APARM       Address of recieving field.
045F E7 00                 STA B 0,X         Move parameter to current parameter list.
0461 08                    INX               Save this pointer along also.
0462 DF A7                 STX   APARM
0464 4A                    DEC A
0465 26 EF                 BNE   P1          Do until all parameters moved.
0467 DE A9                 LDX   TABADR      Get table pointer from parameter list.
0469 D6 A7          PLOOP  LDA B BRESBT      Number of bits per permuted byte.
046B D7 AF                 STA B NBIT        Save it.
046D 7F 00 70              CLR   A3          Clear work byte.
0470 E6 00          MAPIB  LDA B 0,X         Get byte from table.
0472 17                    TBA               Put a copy in A.
```

```
0472 84 F0                AND A  #$F0     Extract seal for source byte.
0474 44                   LSR A           Right justify mask number.
0475 44                   LSR A
0476 44                   LSR A
0477 44                   LSR A
0478 DE 69                LDX    AINPUT   Get input address from parameters.
047A C4 0F                AND B  #$0F     Extract source byte number.
047C BD AD 36             JSR    ADDBX
047F E6 00                LDA B  0,X      Get source byte.
0481 7F 00 71             CLR    TMASK    Prepare mask area.
0484 0D                   SEC             Set carry to be rotated into mask.
0485 76 00 71   SLOOP     ROR    TMASK    Location of next bit is in 1.
0488 4A                   DEC A
0489 2E FA                BGT    SLOOP    Continue shifting mask bit in.
048B D4 71                AND B  TMASK    Use mask to get bit from source byte.
048D C9 FF                ADC B  #$FF     Any one bit will be forced into the carry.
048F 79 00 70             ROL    R0       Move the carry into R0.
0492 DE 6D                LDX    TABADR   Move pointer to next table entry.
0494 08                   INX
0495 DF 6D                STX    TABADR
0497 7A 00 6F             DEC    NBIT     Decrement bit counter.
049A 2E D3                BGT    SHFIN    Continue until finished with current byte.
049C D6 70                LDA B  R0       Get completed work byte.
049E DE 6B                LDX    ARESUD   Get result address.
04A0 E7 00                STA B  0,X      Store work byte.
04A2 08                   INX             Point to next result byte.
04A3 DF 6B                STX    ARESUD
04A5 7F 00 70             CLR    R0       Clear out work byte for next iteration.
04A8 DE 6D                LDX    TABADR   Get current table location.
04AA 7A 00 6B             DEC    NPERM    Decrement byte counter.
04AD 2E B9                BGT    PLOOP    Continue till finished.
04AF 39                   RTS             Return to sender.
                *********************************************
                * SUBROUTINE PERMS:                         *
                *                                           *
                *   FUNCTION: Performs B mapping from E to K1 *
                *             using 8 tables which are compacted *
                *             so that odd-even elements are in *
                *             the same byte.                *
                *                                           *
                * EXTERNAL ROUTINES:                        *
                *     FLEX: ADDBX                           *
                *                                           *
                *********************************************
04B1 C6 08      PERMS     LDA B  #08      Initialize select 8 iteration counter.
04B3 37         PERMS1    PSH B
04B4 DE 99                LDX    KADR     Get current address of K.
04B6 E6 00                LDA B  0,X      Get a byte of K.
04B8 17                   TBA             Put a copy in A.
04B9 C4 1E                AND B  #$1E     Mask bits 4 - 7 to get column no.
04BB 54                   LSR B           Right justify column number.
04BC D7 70                STA B  R0       Save column number for now.
04BE 54                   LSR B           Integer divide col. no. by 2.
04BF 84 21                AND A  #$21     Get row number.
04C1 8B 0F                ADD A  #$0F     Force bit 1 next to bit 3.
04C3 84 F0                AND A  #$F0     Clear right part of byte.
04C5 44                   LSR A           Integer divide row number by 2.
04C6 DE 9D                LDX    SADR     Get beginning address of current B table.
04C8 1B                   ABA             Form byte number within table.
04C9 16                   TAB
04CA BD AD 36             JSR    ADDBX
04CD E6 00                LDA B  0,X      Get table entry.
04CF 76 00 70             ROR    R0       Rotate original col. no. to see if the
04D2 24 07                BCC    GETL     value is on the left or right side.
04D4 7F 00 71             CLR    TMASK    Clear flag byte.
04D7 C4 0F                AND B  #$0F     Get the right value from the table entry.
04D9 20 06                BRA    PSIDE    Go see which side to put it in.
04DB C4 F0      GETL      AND B  #$F0     Get the left value from the table entry.
04DD 86 01                LDA A  #01      Set left side flag.
04DF 97 71                STA A  TMASK    Save it.
04E1 32         PSIDE     PUL A           Get copy of iteration counter.
04E2 36                   PSH A           Restore it to stack.
04E3 46                   ROR A           Rotate rightmost bit into carry.
04E4 24 13                BCC    STL      If 0 then even numbered iteration.
04E6 96 71                LDA A  TMASK    Odd number - if the entry is already
04E8 27 04                BEQ    NSHFR    on the right side then no shift needed.
04EA 54                   LSR B
04EB 54                   LSR B
04EC 54                   LSR B
04ED 54                   LSR B
04EE DE 9B      NSHFR     LDX    K1ADR    Get current byte address.
04F0 EA 00                ORA B  0,X      Place right side of byte in.
04F2 E7 00                STA B  0,X
04F4 08                   INX             Since the byte is now completed,
04F5 DF 9B                STX    K1ADR    increment the result byte pointer.
04F7 20 0C                BRA    STO
04F9 96 71      STL       LDA A  TMASK    Get flag.
04FB 26 04                BNE    NSHFL    If flag is 0 then need to left
04FD 58                   ASL B           justify the table entry to fit
04FE 58                   ASL B           the output.
04FF 58                   ASL B
0500 58                   ASL B
0501 DE 9B      NSHFL     LDX    K1ADR    Get current result byte address.
0503 E7 00                STA B  0,X      Left side -- just store it.
0505 DE 99      STO       LDX    KADR     Increment source data addr pointer.
0507 08                   INX
0508 DF 99                STX    KADR
050A DE 9D                LDX    SADR     Increment table pointer to next table.
050C C6 20                LDA B  #32
050E BD AD 36             JSR    ADDBX
0511 DF 9D                STX    SADR
0513 33                   PUL B           Get iteration counter.
0514 5A                   DEC B
0515 2E 9C                BGT    PERMS1   Keep going.
0517 CE 00 7A             LDX    #K       Reset source data pointer.
051A DF 99                STX    KADR
051C CE 00 82             LDX    #K1      Reset result data pointer.
051F DF 9B                STX    K1ADR
0521 CE 01 D9             LDX    #S       Reset table pointer.
0524 DF 9D                STX    SADR
0526 39                   RTS
                          END

NO ERROR(S) DETECTED
```

```
SYMBOL TABLE:

ADDBX  AD36   AINPUT 0069   APARM  00A7   ARESUD 006B   BADDR  E047
BYTE   E055   C      008E   D      0092   DECR   03ED   E      0184
EXT    0003   GETCHR AD15   GETFIL AD2D   GETL   04DB   GOOM   0407
INEEE  E1AC   INKEY  0338   INPLN  0357   INPMSG 0042   INPUT  0086
INTTAB 00E1   INVTAB 0209   ITER   0333   ITER1  03CB   K      007A
K1     0082   K1ADR  009B   KADR   0099   KEY    009F   KEYMSG 002E
KPARM  00C1   L      0072   LSHIFT 0384   MAIN   0319   MODMSG 0004
MST1   040E   MST2   0424   NBIT   006F   NPERM  006B   NRESET 0067
NSHFL  0501   NSHFR  04EE   OUT2H  E00F   OUT2HS E0CA   OUT4HS E0C8
OUTEEE E1D1   OUTS   E0CC   OUTSHF 0380   P      0189   P1     0456
PARMS  00A9   PCH1A  0121   PCH1B  013D   PDO    0199   PCRLF  AD24
PDATA1 E07E   PERM   044F   PERMS  04B1   PERMS1 04B3   PLOOP  0469
PSIDE  04E1   PSTRNG AD1E   PUTCHR AD18   PX     036D   R      0076
R0     0070   RPTERR AD3F   RSHIFT 0399   S      01D9   SADR   009D
SDIR   0096   SETEXT AD33   SHFIN  0470   SHIFT  0376   SLOOP  0486
SNUM   0097   STL    04F9   STLOC  0065   STO    0505   SUL    0431
TABADR 006D   TMASK  0071   VN     0002   WARMS  AD03
```

```
+++DES
ENTER A 00 FOR ENCRYPT, 01 FOR DECRYPT
00
INPUT KEY (8 BYTES)
0101010101010101
ENTER DATA TO BE ENCODED (8 BYTES)
95F8A5E5DD31D900
80 40 40 40 40 40 40 00
+++DES
ENTER A 00 FOR ENCRYPT, 01 FOR DECRYPT
01
INPUT KEY (8 BYTES)
0101010101010101
ENTER DATA TO BE ENCODED (8 BYTES)
8000000000000000
95 F8 A5 E5 DD 31 D9 00
+++
```

Letters—New Products—Etc.

Alford and Associates

P.O. Box 6743 Richmond, Virginia 23230

phone 804-320-3016

28 November 1979

NEW PRODUCT ANNOUNCEMENT FOR THE M6800 / SS-50 USER

Alford and Associates is pleased to announce the availability of their new SCREDITOR Screen Editing System for operation with Smoke Signal Broadcasting DOS version 5.1X.

The SCREDITOR provides the most flexible and convenient combination of capabilities available in any editing utility today. Dual mode operation is provided, allowing the editing of SOURCE files whose lines must be exactly defined as well as TEXT-typed material, where the need to constantly watch for margin overrun makes line-editors clumsy. In 'TEXT-MODE', part-words are AUTOMATICALLY SHIFTED to the next line, making margin-watching unnecessary. In 'SOURCE-MODE', the SCREDITOR sets a fixed right margin.

The SCREDITOR provides FOURTEEN major edit commands, giving it all the power of the best of line-editors. In addition, in the screen editing mode, TWENTY-TWO screen operators (cursor operations, insertions, deletions, movements, case-changes, etc...) provide a level of control over the edit process that is not matched by any other system in the industry to our knowledge. With editing being done ON-SCREEN, the operator controls the edit process, eliminating costly errors common to line-editors.

The SCREDITOR is designed to operate with either 16 x 64 or the now-available 24 x 80 character memory-mapped displays for the SS-50 buss. Necessary modifications to support a particular memory-mapped display board are minimal, if any.

The manual provided with the software provides a tutorial section to familiarize the operator with the unique features of the SCREDITOR. Complete information to allow the user to modify the package for individual tastes and system considerations is provided. Keyboard definition, system I/O, etc..., are user alterable to meet special requirements.

When compared to other screen-editing software available to the users of microcomputers, the SCREDITOR, priced at $99.95, is one of the best buys available to the M6800 user today. The SCREDITOR may be ordered by phone or mail, and payment may be made by VISA, MASTERCHARGE or by personal check. Check payments will delay the normal three-day shipment by about two weeks on the average. Dealer and OEM inquiries are invited.

A version for the 6809 will be available in the first quarter of 1980.

11/26/79

Mr. Don Williams
'68' Micro Journal
3018 Hamill Road
Hixson, TN 37343

Dear Mr. Williams:

You did not print the truth table of AND, NAND, OR, NOR, Ex.OR in the article, "Logic Gate Tester", in Nov./Dec. 1979 issue on page 33.

Enclosed please find the truth table for your correction.

Sincerely,

S J Houng

S. J. Houng
E. 36 Salmon Street
Spokane, WA 99218

Test Pattern	AND	NAND	OR	NOR	Ex.OR
0 0	0	1	0	1	0
0 1	0	1	1	0	1
1 0	0	1	1	0	1
1 1	1	0	1	0	0
Hex. Value	1	E	7	8	6
Displayed Character					

Test Pattern	Inverter	Buffer
0	1	0
1	0	1
0	1	0
1	0	1
Hex. Value	A	5
Displayed Character		

Clock	D-Input	D-Type Flip-Flop Q	D-Type Flip-Flop $\bar{Q}$	J-K Flip-Flop J=K=1 Q	J-K Flip-Flop J=K=1 $\bar{Q}$
0	0	1	0	0	1
1	0	0	1	0	1
0	1	0	1	1	0
1	1	1	0	1	0
Hex. Value		9	6	3	C
Displayed Character					

November 27, 1979

'68 Micro Journal
3018 Hamill Road
Hixson, Tennessee 37343

Dear Mr. Williams:

While loading and debugging the Christmas File Program submitted by Mr. Paul Phelps in the Nov/Dec issue of '68 Micro Journal, I discovered a small error in the listing. On line 2200: FOR I = 1 to Y + 1 should be FOR I = 1 to Y. After I made this correction the program ran perfectly.

I would also like to take this opportunity to plug what I think is one of the best Floppy Disk Systems available for SS50 buss computers, and this, of course, is Percom's LFD-400" Floppy Disk System. It comes from Percom with "Mindos-Plus X" Percom's named file DOS in ROM.

Sincerely,

DETaylor

D. E. Taylor
Route # 4 - Box 196
Seneca, S.C.

DET/PC

Inventory Problems?

Are you having trouble keeping the right nuts and bolts in stock? Since even a simple mistake can cost you time and money, a good inventory system should do more than just count parts. It should tell you exactly what you need, when you need it, where to get it, and how much it will cost.

The MSI Inventory System Seven enables you to maintain a versatile data base for controlling inventory. It lists part number, description, quantity on hand, vendor, cost, selling price, optional pricing, usage levels for previous month, present month, and year-to-date, and much more.

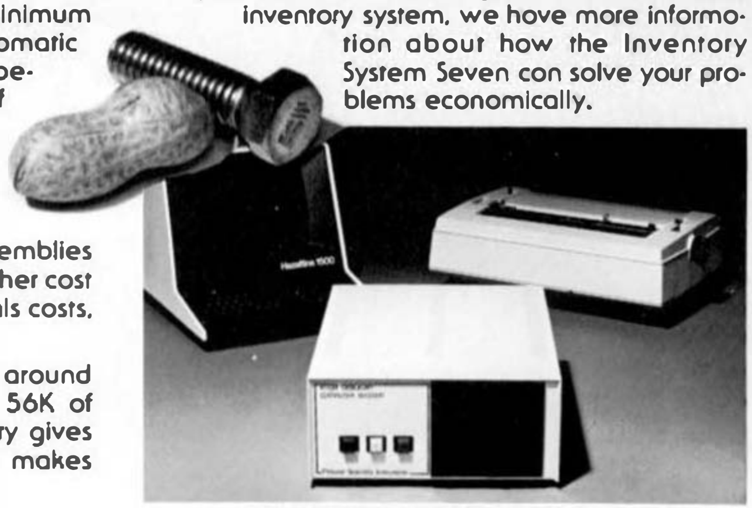
When quantity on hand items reach minimum levels, the System Seven compiles an automatic reorder list. This list can be generated by specific vendor as well as a complete listing of all materials to be ordered.

In addition to the item listing, the Inventory System Seven "bill of materials" provides you with a complete inventory of items used in the manufacture of subassemblies and complete products. It also contains other cost items such as labor costs, total raw materials costs, and miscellaneous costs.

The MSI Inventory System Seven is built around the versatile MSI 6800A Computer with 56K of RAM. An integral dual mini-floppy memory gives you an additional 630K of memory and makes inventory control fast and efficient. The System Seven will interface with any industry standard CRT, and you have the option of both a "daisy wheel" word processor for high quality document preparation and a dot matrix printer for high speed production.

The System Seven can be expanded to handle all your data processing needs or you can select one of nine other MSI systems now available for business, industrial, scientific, educational, and personal applications.

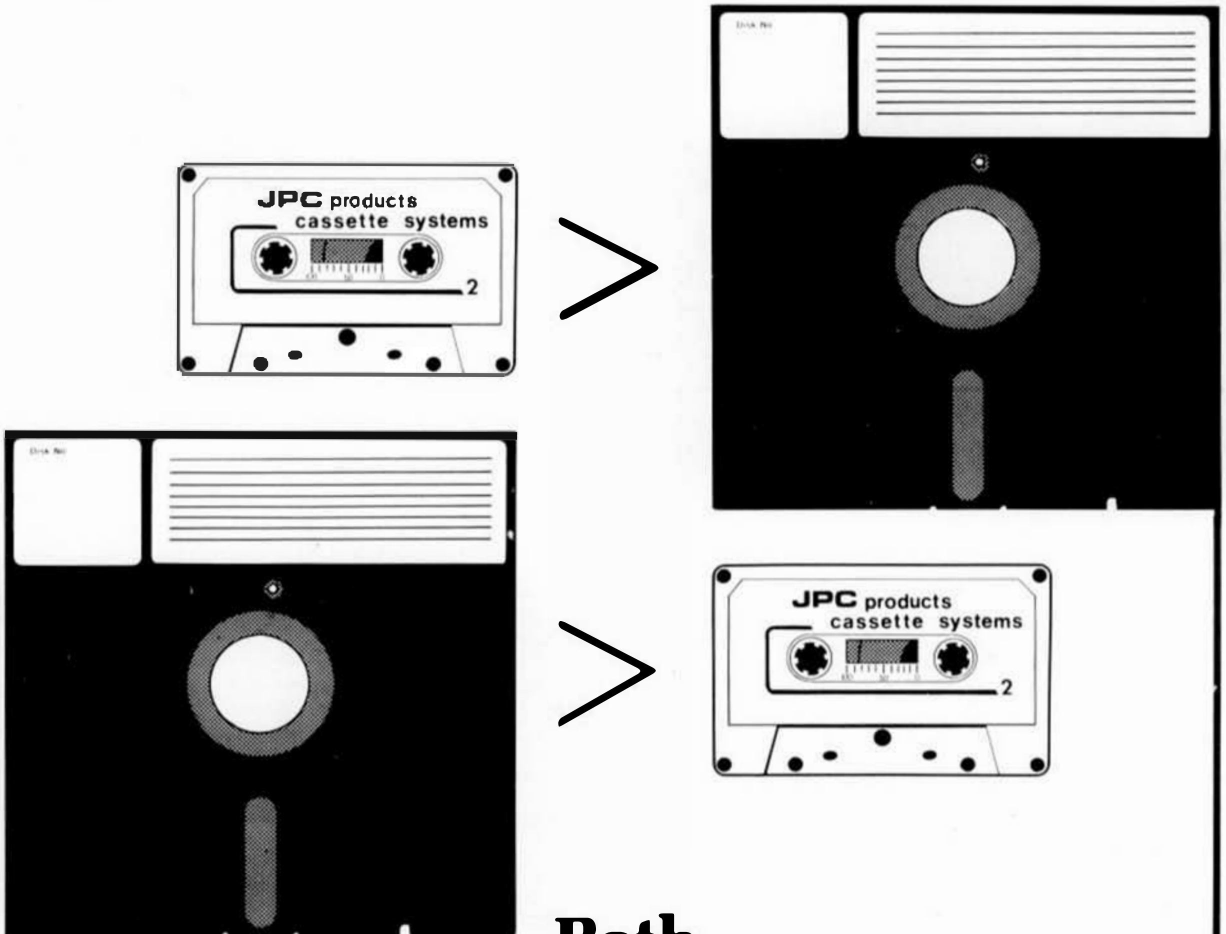
If you need more than just a nuts and bolts inventory system, we have more information about how the Inventory System Seven can solve your problems economically.



MSI Inventory System Seven

MSI Midwest Scientific

220 W. Cedar, Olathe, Kansas 66061, (913) 764-3273
TWX 910 749 6403 (MSI OLAT), TELEX 42525 (MSI A OLAT)



Both Statements are True.

Cassettes are greater than floppy disks. Their load capacity holds as much data as twelve 5" floppies.

And floppy disks are greater than cassettes. They are faster. And their cost is much greater. Up to 10 times as much. Plus, they take more space and fuss in storage.

Therefore:

Get a TC-3 Cassette Interface Kit. Only $59.95. It will allow your computer to read and write programs on standard audio cassettes. The reliable, easy-to-assemble board plugs into one I/O slot of an SWTPC Motherboard so it reads and writes at a rate of 4800 BAUD. (Load 4K in 8 sec!)

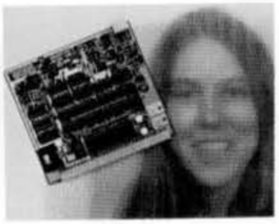
As a primary system or as a disk back-up system, it can't be beat. This statement is true.

Terms: Check, Money Order, MC or Visa. $3 shipping.

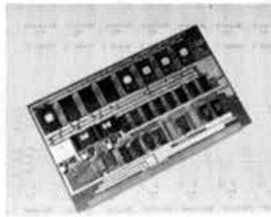
JPC products

P.O. Box 5615, Albuquerque, NM 87185 Order Phone 505-294-4623

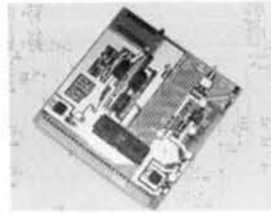
See Gimix Ad on page 3



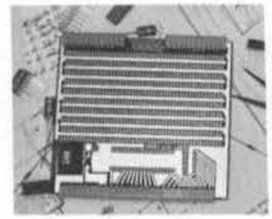
DS-68 DIGISECTOR



PSB-08 PROM SYSTEM BOARD



B-08 EPROM PROGRAMMER



UIO UNIVERSAL I/O BOARD

INNOVATIVE PRODUCTS FOR 6800 USERS

DS-68 DIGISECTOR is a random access video digitizer featuring 256 × 256 picture element scan and 64 levels of grey scale, with conversion times as low as 3 microseconds per pixel. It accepts either interlaced (NTSC) or non-interlaced (industrial) video input. Use it for computer portraiture, moving target indicators, precision security systems, fast to slow scan conversion. . . with clever software, the Digisector can read just about anything. Truly a professional tool at a price you can afford. $169.95

DM-85 DISK MIXER is an add-on board for the Smoke Signal Broadcasting BFD-68A Disk Controller which allows operation of both 8" and 5" drives. Controller mode (8" or 5") is selected on a drive-by-drive basis, so any mix of 5" and 8" drives is allowable. The 2" × 3" PC board mounts inconspicously on the back of the BFD-68A. Its operation is completely transparent to software. An oscilloscope is required for the setup procedure. Kit Price: $39.95

B-08 2708 EPROM PROGRAMMER is a compact unit that fits in the 6800's I/O slot. A safety switch and LED indicator provide control over the high programming voltage generated on board. An industrial quality Textool socket and extended board height allow effortless PROM insertion and retrieval. Fully commented source listings of U2708 is included in the Owner's Manual. $99.95

U2708 utility for testing, burning, verifying and copying 2708s in EPROM. $29.95

M6809 EMULATOR is a machine language program that will emulate all of the functions of the Motorola 6809 third generation microprocessor. Developed for use on any 6800 system, the program allows software development and debugging. The 3K byte program is complete with a 6809 mini-monitor and single-step trace routines. Fully commented source listing included. Specify Smoke Signal Broadcasting or FLEX™ disk, or KCS cassette. $49.95

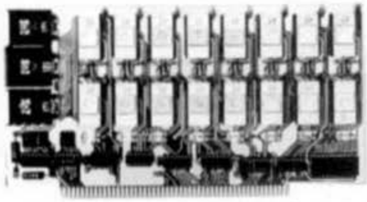
PSB-08 PROM SYSTEM BOARD features 1K of high speed, low-power RAM and space for up to 8 2708 EPROMs, both DIP-switch addressable to start on any 8K boundary in memory. The exclusive I/O select feature allows you to move I/O locations up to any unused 1K block in the EPROM memory space. This permits memory expansion to a full 56K of contiguous user RAM. $119.95

UIO UNIVERSAL I/O BOARD helps you with your custom interfaces. It has space for a 40-pin wire wrap socket into which you may plug any of Motorola's 40 or 24-pin interface chips. All data and control lines are connected to the appropriate edge connector pins. All other bus connections are brought out to a 16-pin socket pad. +5 volt regulator and all Molex connectors are provided; regulated +5 and ground are bused among the locations for up to 35 14-pin ICs. $24.95

THE MICRO WORKS P.O. BOX 1110, DEL MAR, CA 92014 714-942-2400

DIGITAL RESEARCH COMPUTERS
(214) 494-1505

16K EPROM CARD-S 100 BUSS



$59.95 KIT

FIRST TIME OFFERED!
BLANK PC BOARD - $28

USES 2708's!

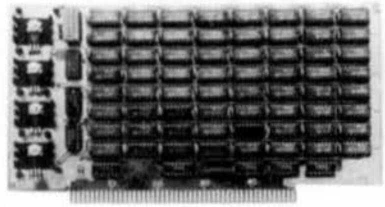
Thousands of personal and business systems around the world use this board with complete satisfaction. Puts 16K of software on line at **ALL TIMES!** Kit features a top quality solder masked and silk-screened PC board and first run parts and sockets. Any number of EPROM locations may be disabled to avoid any memory conflicts. Fully buffered and has WAIT STATE capabilities.

OUR 450 NS 2708'S ARE $8.95 EA. WITH PURCHASE OF KIT

ASSEMBLED AND FULLY TESTED ADD $25

8K LOW POWER RAM KIT-S 100 BUSS

SALE



PRICE CUT!

$119.50 KIT

21L02
(450 NS RAMS!)

Thousands of computer systems rely on this rugged, work horse, RAM board. Designed for error-free, NO HASSLE, systems use.

ASSEMBLED AND FULLY BURNED IN ADD $30

ALL ASSEMBLED BOARDS ARE TESTED AT 4MHZ.

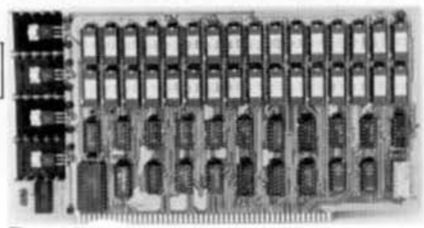
Blank PC Board w/Documentation $29.95
Low Profile Socket Set 13.50
Support IC's (TTL & Regulators) $9.75
Bypass CAP's (Disc & Tantalums) $4.50

16K STATIC RAM KIT-S 100 BUSS

PRICE CUT!

$259 KIT

FOR 4MHZ ADD $25



KIT FEATURES
1. Addressable as four separate 4K Blocks
2. ON BOARD BANK SELECT circuitry (Cromemco Standard!) Allows up to 512K on line!
3. Uses 2114 (450NS) 4K Static Rams
4. ON BOARD SELECTABLE WAIT STATES
5. Double sided PC Board, with solder mask and silk screened layout. Gold Plated contact fingers
6. All address and data lines fully buffered
7. Kit includes ALL parts and sockets
8. PHANTOM is jumpered to PIN 67
9. LOW POWER under 1.5 amps TYPICAL from the +8 Volt Buss
10. Blank PC Board can be populated as any multiple of 4K

BLANK PC BOARD W/DATA-$33
LOW PROFILE SOCKET SET-$12
SUPPORT IC'S & CAPS-$19.95
ASSEMBLED & TESTED-ADD $30

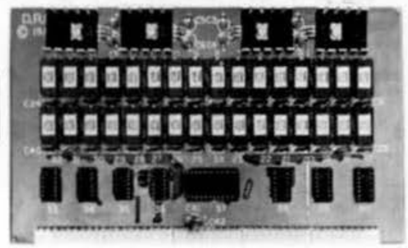
OUR #1 SELLING RAM BOARD!

16K STATIC RAM SS-50 BUSS

PRICE CUT!

$249 KIT

FULLY STATIC AT DYNAMIC PRICES



FOR SWTPC 6800 BUSS!

ASSEMBLED AND TESTED - $30

KIT FEATURES
1. Addressable on 16K Boundaries
2. Uses 2114 Static Ram
3. Runs at Full Speed
4. Double sided PC Board. Solder mask and silk screened layout. Gold fingers
5. All Parts and Sockets included
6. Low Power: Under 1.5 Amps Typical

BLANK PC BOARD—$28   COMPLETE SOCKET SET—$12
SUPPORT IC'S AND CAPS—$19.95

PROC. TECH. QUITS THE MICROPROCESSOR BUSINESS!
FACTORY CLOSE OUT - SPECIAL PURCHASE!
#16KRA

16K S-100 Dynamic Ram Board

$149.95

We purchased the remaining inventory of PT's popular 16K Ram Board when they recently closed their plant. Don't miss the boat! These are brand new, fully tested, ASSEMBLED and ready to go. All are sold with our standard 90 day limited warranty!!

Orig. $429 each! 72 Page Full Manual. Included Free!

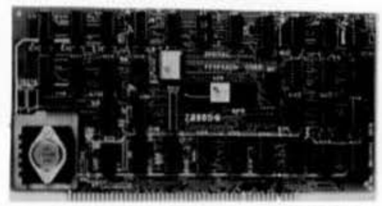
S-100 Z80 CPU CARD

Perfect for OEM's

$159.95

WIRED! NOT A KIT!

4 MHZ



ASSEMBLED AND TESTED! READY TO USE! Over 3 years of design efforts were required to produce a **TRUE** S-100 Z80 CPU at a genuinely bargain price!

BRAND NEW!

FEATURES
- ★ 2 or 4 MHZ Operation
- ★ Generates MWRITE, so no front panel required
- ★ Jump on reset capability
- ★ 8080 Signals emulated for S-100 compatability
- ★ Top Quality PCB, Silk Screened, Solder Masked, Gold Plated Contact Fingers

Z-80 PROGRAMMING MANUAL

By MOSTEK, or ZILOG. The most detailed explanation ever on the working of the Z-80 CPU CHIPS. At least one full page on each of the 158 Z-80 instructions. A **must** reference manual for any user of the Z-80. 300 pages.

$12.95

NEW! G.I. COMPUTER SOUND CHIP

AY3-8910. As featured in July, 1979 BYTE! A fantastically powerful Sound & Music Generator. Perfect for use with any 8 Bit Microprocessor. Contains: 3 Tone Channels, Noise Generator, 3 Channels of Amplitude Control, 16 Bit Envelope Period Control, 2-8 Bit Parallel I/O, 3 D to A Converters, plus much more! All in one 40 Pin DIP. Super easy to interface to the S-100 or other busses.

SPECIAL OFFER: $14.95 each   Add $3 for 64 page Data Manual.

SALE! LOW POWER - 250NS 2114 RAM SALE!

8 FOR $55

4K STATIC RAM'S. MAJOR BRAND. NEW PARTS.
These are the most sought after 2114's. LOW POWER and 250NS FAST.
SPECIAL SALE: $7.00 ea. or 8 For $55

Digital Research Computers
(OF TEXAS)
P.O. BOX 401565 • GARLAND, TEXAS 75040 • (214) 494-1505

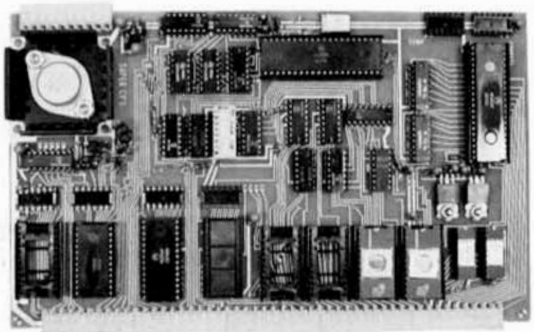
TERMS: Add $1.00 postage, we pay balance. Orders under $15 add 75¢ handling. No C.O.D. We accept Visa and MasterCharge. Tex. Res. add 5% Tax. Foreign orders (except Canada) add 20% P & H. 90 Day Money Back Guarantee on all items.

NOT ASSOCIATED WITH DIGITAL RESEARCH OF CALIFORNIA, THE SUPPLIERS OF CPM SOFTWARE.

NOW INDUSTRIAL QUALITY AT LOW COST

FROM THOMAS INSTRUMENTATION

Industrial system boards are now available separately for OEM, prototyping or hobbyist applications. Shipped from stock, these are the same quality cards used in monitors and machine tool controls designed for GM, LTI, and General Electric. All cards are SS-50 buss compatible and are suitable for dedicated applications. The CPU card and the Video RAM Card may be combined on a TI backplane as a stand-alone micro — ideal for prototypes or hobbyists. ATTENTION OEM's: If you have a control data acquisition, monitoring, or other microprocessor application. Check with TI for more information about custom software design for the TI CPU or any other 6800 series system. TI also has non-SS-50 buss single board 6800 systems.



SS-50 SUPER CPU

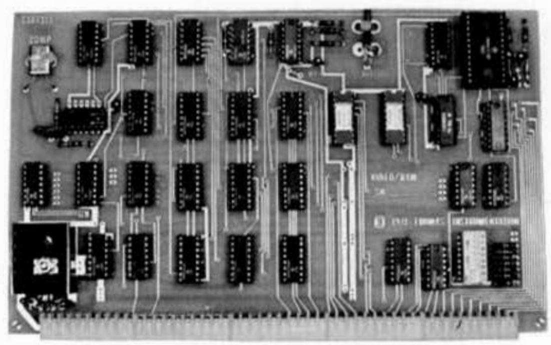
- SS-50 or stand alone computer
- 1K of RAM at $A000 I/O on board at $A400 (Relocatable)
- 2K Monitor (Mikbug compatible) in 2708 EPROM
- 2 8 bit parallel ports with 2 control bits and power
- RS-232 ACIA port, 2nd TTL ACIA optional
- 3 16 bit counter/timers (expandable to 6 add 2nd 6840)
- 128 byte RAM at 0000 is jumper selectable
- Battery back-up for 32 bytes of RAM
- Plug back to back with Video RAM for Stand alone micro or customized smart terminal

ASSEMBLED $195.00
CARD AND DOCUMENTATION $49.00

SS-50 VIDEO RAM

- Fully synchronous operation — No jitter
- 7 by 9 Characters Programmable reverse video
- Full 128 Character ASCII set
- 1K of memory can be mapped to any 1K boundary
- Full documentation includes software (Replaces OUTEE)

ASSEMBLED $149.00
CARD, CRYSTAL and DOCUMENTATION $39.00



TI SS-50 Wire-Wrap Card	**24.00**
TI SS-50 Parallel I/O Card	**95.00**
Card only	**35.00**
GIMIX 16K Static RAM w/Soft addressing	**368.00**
GIMIX 16 RAM without Soft addressing	**298.00**
3, 4, 7 SLOT Backplanes (per slot)	**4.00**

TI cards available from stock

THOMAS INSTRUMENTATION

168-8th Street, Avalon, N.J. 08202
Phone (809) 967-4280

DEALERS FOR GIMIX SWTPC SSB

CALL FOR DEALER, OEM. AND QUANTITY PRICES

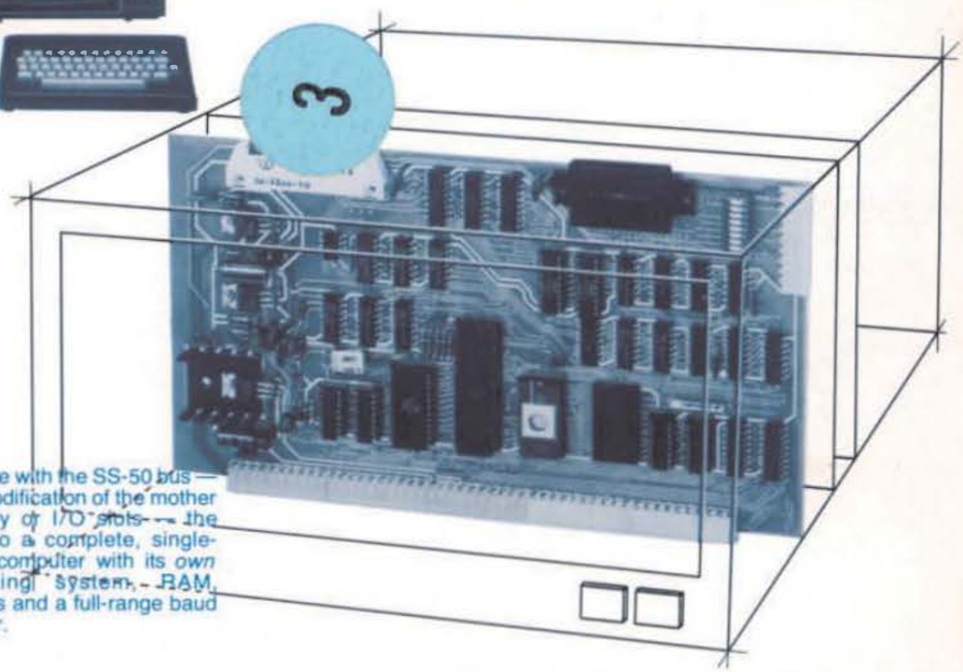
'68' Micro Journal
3018 Hamill Rd.
Hixson, TN 37343

Second Class Postage Paid
At Chattanooga, TN
ISSN 0194-5025

000422 A/E
MR. MICKEY FERGUSON
BOX 708
TRENTON GA 30752
MJ

6809 PROCESSING POWER!

The Percom SBC/9™. Only $199.95.



Fully compatible with the SS-50 bus — requiring *no* modification of the mother board, memory or I/O slots — the SBC/9™ is also a complete, single-board control computer with its *own* ROM operating system, RAM, peripheral ports and a full-range baud clock generator.

Make the SBC/9™ the heart of your computer and put to work the most outstanding microprocessor available, the 6809.

the Mighty 6809

Featuring more addressing modes than any other eight-bit processor, position-independent coding, special 16-bit instructions, efficient argument-passing calls, autoincrement/autodecrement and more, it's no wonder the 6809 has been called the "programmers dream machine."

Moreover, with the 6809 you get a microprocessor whose programs typically use only one-half to two-thirds as much RAM space as required for 6800 systems, and run faster besides.

And to complement the extraordinary 6809, the Percom design team has developed PSYMON™, an extraordinary 6809 operating system for the SBC/9™.

PSYMON™ — Percom SYstem MONitor

Although PSYMON™ includes a full complement of operating system commands and 15 externally callable utilities, what really sets PSYMON™ apart is its easy hardware adaptability and command extensibility.

For hardware interfacing, you merely use simple, specific device driver routines that reference a table of parameters called a Device Control Block (DCB). Using this technique, interfacing routines are independent of the operating system.

The basic PSYMON™ command repertoire may be readily enhanced or modified. When PSYMON™ first receives system control, it initializes its RAM area, configures its console and then 'looks ahead' for an optional second ROM which you install in a socket provided on the SBC/9™ card. This ROM contains your own routines that may alter PSYMON™ pointers and either subtly or radically modify the PSYMON™ command set. If a second ROM is not installed, control returns immediately to PSYMON™.

- Provision for multi-address, 8-bit bidirectional parallel I/O data lines for interfacing to devices such as an encoded keyboard.
- A serial interface Reader Control output for a cassette, tape punch/reader or similar device.
- An intelligent data bus: multi-level data bus decoding that allows multiprocessing and bus multiplexing of other bus masters.
- Extended address line capability — accommodating up to 16 megabytes of memory — that does not disable the on-board baud rate clock or require additional hardware in I/O slots.
- On-board devices which are fully decoded so that off-card devices may use adjoining memory space.
- Fully buffered address, control and data lines.

The SBC/9™, complete with PSYMON™ in ROM, 1K of RAM and a comprehensive users manual™ costs just $199.95.

™ trademark of Percom Data Company, Inc.

PERCOM

PERCOM DATA COMPANY, INC.
211 N. KIRBY GARLAND, TEXAS 75042
(214) 272-3421

Percom *'peripherals for personal computing'*

To place an order or request additional literature call toll-free 1-800-527-1592. For technical information call (214) 272-3421. Orders may be paid by check, money order, COD or charged to a VISA or Master Charge account. Texas residents must add 5% sales tax.

PRICES AND SPECIFICATIONS SUBJECT TO CHANGE WITHOUT NOTICE